Developer's Guide
SAP NetWeaver 7.0 SPS12

Enterprise Data Warehousing

Document Version 4.00 – May 2007

© Copyright 2007 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.
IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, Informix, i5/OS, POWER, POWER5, OpenPower and PowerPC are trademarks or registered trademarks of IBM Corporation.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.
Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.
Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies.

Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

Disclaimer

Some components of this product are based on Java™. Any code change in these components may cause unpredictable and severe malfunctions and is therefore expressively prohibited, as is any decompilation of these components.

Any Java™ Source Code delivered with this product is only to be used by SAP's Support Services and may not be modified or altered in any way.

Documentation on SAP Help Portal

You can find this documentation at help.sap.com/nw70

Typographic Conventions

Type Style	Represents
Example Text	Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options.
	Cross-references to other documentation.
Example text	Emphasized words or phrases in body text, graphic titles, and table titles.
EXAMPLE TEXT	Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, SELECT and INCLUDE.
`Example text`	Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
`Example text`	Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.
`<Example text>`	Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.
`EXAMPLE TEXT`	Keys on the keyboard, for example, F2 or ENTER.

Icons

Icon	Meaning
	Caution
	Example
	Note
	Recommendation
	Syntax

Contents

1 ENTERPRISE DATA WAREHOUSING 1

2 GETTING INVOLVED 2

2.1 The Data Warehouse Concept 2
2.1.1 Using a Data Warehouse 2
2.1.2 Architecture of a Data Warehouse 2
2.1.3 Enterprise Data Warehouse (EDW) 4
2.1.4 Building and Running a Data Warehouse 4
2.2 Working with the Development Environment 5
2.2.1 Data Warehousing Workbench 6

3 GO AND CREATE 9

4 CORE DEVELOPMENT TASKS 10

4.1 Developing Business Logic 10
4.1.1 Recommendations for Modeling Layers 11
4.1.2 Recommendations for Modeling the Data Flow 12
4.1.3 Recommendations for Modeling InfoCubes 15
4.1.4 Recommendations for Modeling InfoSets 15
4.1.5 Recommendations for Modeling MultiProviders 16
4.1.6 Modeling 17
4.1.7 Transformation 232
4.1.8 Data Transfer Process 320
4.1.9 Scheduler 336
4.1.10 Process Chain 375
4.1.11 Non-Cumulatives 418
4.1.12 Installing BI Content 435
4.1.13 Customer and Partner Content 444
4.2 Developing Persistency 507
4.2.1 Differences Between Data Stores 507
4.3 Using Connectivity and Interoperability 508
4.3.1 Data Acquisition 509
4.3.1.2.5 Editing DataSources from SAP Source Systems in BI 528
4.3.2 Real-Time Data Acquisition 630
4.3.3 Open Hub Destination 651
4.3.4 ICF Services in BI 656
4.4 Considering Performance 658
4.4.1 DB Storage Parameters 659
4.4.2 Partitioning 660
4.4.3 Clustering 662
4.4.4 Performance Optimization for DataStore Objects 663
4.4.5 Performance Optimization with Aggregates 665
4.4.6 Performance Optimization with BI Accelerator 700
4.4.7 Compressing InfoCubes 735
4.4.8 InfoCube Performance 736

THE BEST-RUN BUSINESSES RUN SAP

4.4.9 Information Lifecycle Management ... 738
4.5 Applying Extensibility ... 768
4.6 Enabling Globalization ... 768
4.6.1 Translating Text for BI Objects ... 768

5 ENSURING QUALITY ... 773

5.1 Testing ... 773
5.1.1 BI Content Analyzer ... 774
5.1.2 Analysis and Repair Environment ... 792
5.2 Debugging ... 796
5.2.1 Update Simulation in the Extraction Monitor ... 797

6 REFERENCE ... 798

6.1 Interface Overview ... 798
6.1.1 DB Connect ... 799
6.1.2 UD Connect ... 800
6.1.3 BI Service API ... 800
6.1.4 File Interface ... 801
6.1.5 Web Service for Staging ... 801
6.1.6 XI Integration ... 802
6.1.7 Staging BAPIs ... 802
6.1.8 DataStore Object APIs ... 802
6.1.9 Interface for Real-Time InfoCubes ... 803
6.1.10 APIs for Master Data ... 804
6.1.11 APIs for Hierarchies ... 804
6.1.12 VirtualProviders with Function Modules ... 805
6.1.13 Open Hub Destination ... 805
6.1.14 Open Hub Destination APIs ... 806
6.1.15 Data Mart Interface ... 807
6.2 Checklist for Developing BI Objects ... 807

1 Enterprise Data Warehousing

Purpose

Various BI interfaces and tools are available if you want to modify the *Enterprise Data Warehousing* scenario.

Advantages for Application Development

- With BI, you can develop your own data models that best represent the processes within your enterprise.
- Customers and partners of BI can develop their own content and deliver it to their business units. You can also make limited modifications and changes to BI Content.
- BI uses heterogeneous interfaces to provide an open architecture, in various directions: You can use third-party tools to extract data from various source systems into a BI system, and use third-party tools to distribute this data into further non-SAP systems.
- BI allows you to implement routines for transferring data.
- With the SAP enhancement concept, you can make enhancements to the standard in the BI system. In the BI system, customer exits and BAdIs are used in open hub destinations, the formula editor and documents to enable enhancements.
- Services that are based on the SAP NetWeaver Internet Communication Framework (ICF) are delivered with BI. The HTTP services of BI allow you to display or swap BI metadata using a URL. You can also display user-specific documentation on metadata.

Prerequisites

Area	Prerequisites
Extraction of data and metadata to a BI system using third-party tools	ABAP proficiency
Distribution of data to non-SAP systems	ABAP proficiency
Modeling BI objects	ABAP proficiency
Developing customer or partner content	ABAP proficiency, ABAP development license
Implementing routines	ABAP proficiency
Enhancements using function exits and BAdIs	ABAP proficiency
ICF services	ABAP proficiency

How to Use This Documentation

If you have no experience of working with *Enterprise Data Warehousing*, start with the following sections:

Working with the Development Environment [Page 5]

Go and Create [Page 9]

These sections contain information about the basic principles, as well as introductory tutorials.

If you already have experience of working with *Enterprise Data Warehousing*, you can use the other sections. These specifically aim to cover the details.

2 Getting Involved

This section provides an overview of the concepts and the development environment.

2.1 The Data Warehouse Concept

The following documentation describes the data warehouse concept. As well as general information about the architecture and uses of a data warehouse, it shows the concrete implementation of the concept within SAP NetWeaver BI.

2.1.1 Using a Data Warehouse

The reporting, analysis, and interpretation of business data is of central importance to a company in guaranteeing its competitive edge, optimizing processes, and enabling it to react quickly and in line with the market.

Company data is usually spread across several applications that are used for entering data. Analyzing this data is not only difficult because it is spread across several systems but because the data is saved in a form that is optimized for processing, not analysis. Data analysis represents additional system load which affects operative data processing. Furthermore, the data has come from heterogeneous applications and is therefore only available in heterogeneous formats which must first be standardized. The applications also only save historic data to a limited extent. This historic data can be important in analysis.

Therefore separate systems are required for storing data and supporting data analysis requirements. This type of system is called a data warehouse.

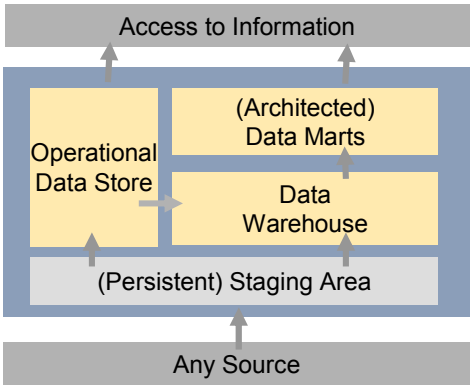
A data warehouse serves to integrate data from heterogeneous sources, transform, consolidate, clean up and store this data, and stage it efficiently for analysis and interpretation purposes.

2.1.2 Architecture of a Data Warehouse

There are many different definitions of a data warehouse. However, they all favor a layer-based architecture.

Data warehousing has developed into an advanced and complex technology. For some time it was assumed that it was sufficient to store data in a star schema optimized for reporting. However this does not adequately meet the needs for consistency and flexibility in the long run. Therefore data warehouses are now structured using a layer architecture. The different layers contain data in differing levels of granularity. We differentiate between the following layers:

- Persistent staging area
- Data warehouse
- Architected data marts
- Operational data store



Persistent Staging Area

After it is extracted from source systems, data is transferred to the entry layer of the data warehouse, the persistent staging area (PSA). In this layer, data is stored in the same form as in the source system. The way in which data is transferred from here to the next layer incorporates quality-assuring measures and the transformations and clean up required for a uniform, integrated view of the data.

Data warehouse

The result of the first transformations and clean up is saved in the next layer, the data warehouse. This data warehouse layer offers integrated, granular, historic, stable data that has not yet been modified for a concrete usage and can therefore be seen as neutral. It acts as the basis for building consistent reporting structures and allows you to react to new requirements with flexibility.

Architected Data Marts

The data warehouse layer provides the most multidimensional analysis structures. These are also called architected data marts. This layer satisfies data analysis requirements. Data marts are not necessarily to be equated with the terms summarized or aggregated; here too you find highly granular structures but they are focused on data analysis requirements alone, unlike the granular data in the data warehouse layer which is application neutral so as to ensure reusability.

The term “architected“ refers to data marts that are not isolated applications but are based on a universally consistent data model. This means that master data can be reused in the form of *Shared* or *Conformed Dimensions*.

Operational Data Store

As well as strategic data analysis, a data warehouse also supports operative data analysis by means of the operational data store. Data can be updated to an operational data store on a continual basis or at short intervals and be read for operative analysis. You can also forward the data from the operational data store layer to the data warehouse layer at set times. This means that the data is stored in different levels of granularity: while the operational data store layer contains all the changes to the data, only the days-end status, for example, is stored in the data warehouse layer.

The layer architecture of the data warehouse is largely conceptual. In reality the boundaries between these layers are often fluid; individual data memory can play a role in two different layers. The technical implementation is always specific to the organization.

2.1.3 Enterprise Data Warehouse (EDW)

The type of information that a data warehouse should deliver is largely determined by individual business needs. In practice this often results in a number of isolated applications which are referred to as *silos* or *stove pipes*. To avoid isolated applications, a comprehensive, harmonized data warehouse solution is often favored; the enterprise data warehouse.

An enterprise data warehouse (EDW) is a company-wide data warehouse that is built to include all the different layers. An organization-wide, single and central data warehouse layer is also referred to as an EDW.

An enterprise data warehouse has to provide flexible structures and layers so that it can react quickly to new business challenges (such as changed objectives, mergers, acquisitions).

2.1.4 Building and Running a Data Warehouse

The structure and running of a data warehouse in general, and an enterprise data warehouse in particular, is highly complex and cannot be tackled without the support of adequate tools. Business Intelligence in SAP NetWeaver offers an integrated solution that represents the entire data warehouse process from extraction, to the data warehouse architecture, to analysis and reporting.

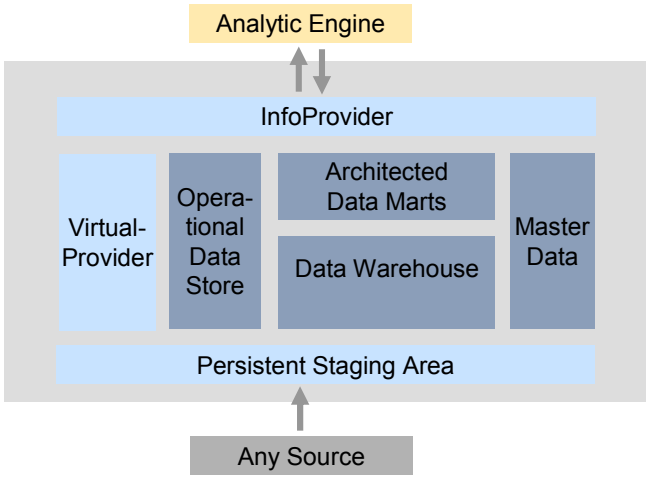
Data Warehousing as an Element of Business Intelligence in SAP NetWeaver Provides:

- **Data staging:**
  - Extraction, transformation, loading (ETL) of data: All data sources can be accessed by means of extraction in the background (via JDBC, file, XMLA, ODBO, ..). Extractors are delivered for SAP applications or can be generated. The standard applications of other providers can be accessed by integrating the ETL tools of non-SAP providers.
  - Real-time data warehousing: Event-near availability of data in the operational data store can be realized using real-time data acquisition technology.
  - Remote data access: Data can be accessed without being saved in the BI system using VirtualProviders (see below).
- **Modeling a layer architecture:** InfoCubes support the modeling of star schemas (with one large fact table in the center and several surrounding dimension tables) in the architected data mart layer. VirtualProviders allow you to access source data directly. InfoCubes can be combined in virtual star schemas (MultiProvider) using *Shared* or *Conformed Dimensions* (master data tables).

  The persistent staging area, data warehouse layer and operational data store are built from flat stores, the DataStore objects.

  InfoObjects (characteristics and key figures) form the basis of the InfoCube or DataStore object description. You ensure vertical consistency by using the same InfoObjects in the various layers and thus avoid the interface problems that can arise if you use heterogeneous tools to build the layers.
- **Transformation:** Transformation rules serve to clean up and consolidate data.

- **Modeling the data flow:** Data transfer processes serve to transfer the data to the different stores. You use process chains to schedule the data processing and observe it using a monitor.
- **Staging data for analysis:** You can define queries based on any InfoProvider using the Business Explorer. BEx queries form the basis of the applications that are available to users in the portal or based on Microsoft Excel.



2.2 Working with the Development Environment

Purpose

The BI system provides heterogeneous development environments.

Area	Development Environment
Extraction of data and metadata to a BI system using third-party tools	ABAP Workbench tools
Distribution of data to non-SAP systems	ABAP Workbench tools
Modeling BI Objects	Data Warehousing Workbench in the BI system
Developing Customer or Partner Content	Data Warehousing Workbench in the BI system
Enhancements Using Routines	Routine Editor of the Data Warehousing Workbench in the BI system
Customer Exits and BAdIs	ABAP Workbench tools
ICF Services	Internet Communication Framework (ICF)

For more information about the Data Warehousing Workbench, see Data Warehousing Workbench [Page 6].

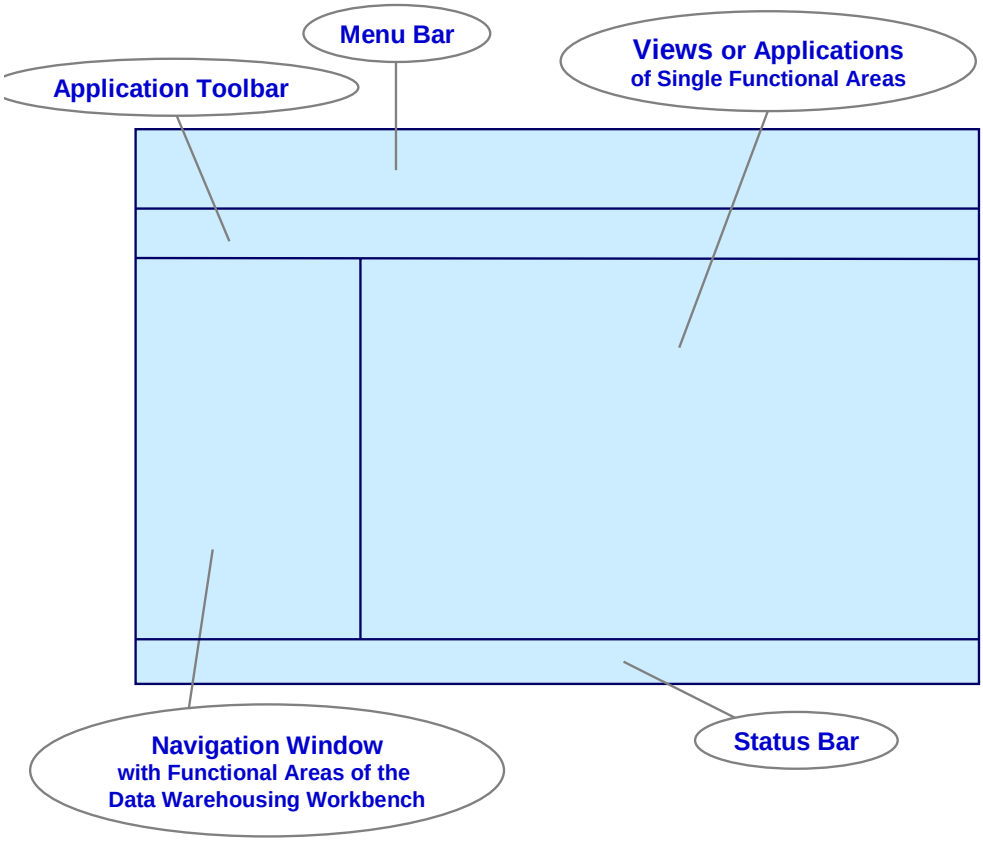
2.2.1 Data Warehousing Workbench

Purpose

The Data Warehousing Workbench (DWB) is the central tool for performing tasks in the data warehousing process. It provides data modeling functions as well as functions for control, monitoring and maintenance of all processes in SAP NetWeaver BI related to with data procurement, data retention, and data processing.

Structure of the Data Warehousing Workbench

The following figure shows the structure of the Data Warehousing Workbench:



Navigation Window with Functional Areas of the Data Warehousing Workbench

When you call the Data Warehousing Workbench, a navigation window appears on the left of the screen. You can open the individual functional areas of the Data Warehousing Workbench using the application toolbar in the navigation window. The applications that are available in these areas are displayed in the navigation window; in the modeling functional area, the possible views of the object trees are displayed.

Object Trees or Applications of the Individual Functional Areas

If object trees or applications are assigned to a functional area in the navigation window, they can be called by simply clicking in the right part of the screen.

Application Toolbar

The toolbar of the Data Warehousing Workbench contains a pushbutton for showing or hiding the navigation window as well as pushbuttons that are relevant in the context of the individual functional areas and applications.

Menu Bar

The functions that can be called with the menu bar of the Data Warehousing Workbench are independent of the functional areas.

Status Bar

Information, warnings, and error messages are displayed in the status bar.

Features

Functional Areas of the Data Warehousing Workbench

Functional area	Documentation
Modeling	Modeling [Page 17]
Administration	Administration [External] and Monitoring and Troubleshooting [External]
Transport connection	Transport System [External]
Documents	Documents [External]
Business Content	BI Content (Versions) [External]
Translation	Translating Text for BI Objects [Page 768]
Metadata Repository	Metadata Repository [External]

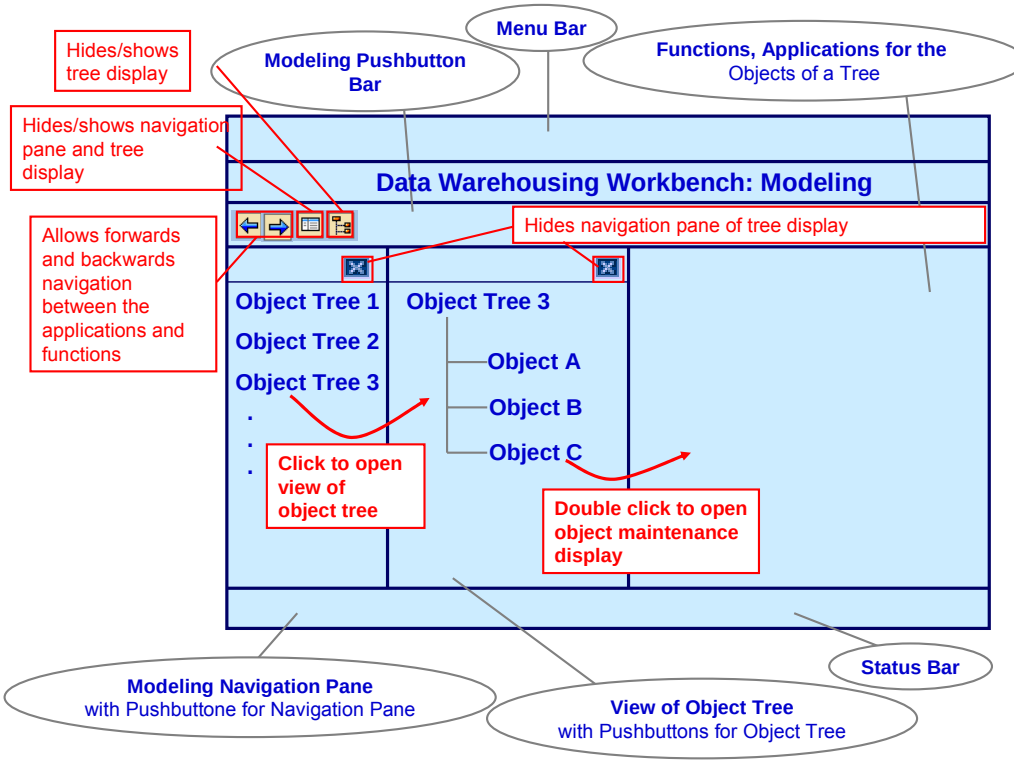
2.2.1.1 Data Warehousing Workbench - Modeling

Purpose

In the *Modeling* functional area of the Data Warehousing Workbench, you can display BI objects and the corresponding dataflow in a structured way in object trees. You can create new objects, call applications and functions for objects and define the dataflow for the objects.

Structure of Data Warehousing Workbench: Modeling

The following graphic illustrates the structure of the *Data Warehousing Workbench: Modeling*:



The *Modeling* functional area consists of various screen areas. As well as the menu, title and status bars, the modeling screen contains the following four screen areas:

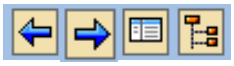
- Modeling pushbutton bar
- Navigation pane in the left-hand area of the screen
- View of selected object tree in right-hand area of screen and, with open applications, the middle area of the screen
- Open application in the right-hand area of the screen

Basic Navigation Options

From the navigation pane, you can click on an entry in the object tree list to open the view of this object tree.

In the object tree, you can expand the nodes to navigate in the objects. You can jump to the corresponding application, usually the object maintenance display, by double clicking on the object name in the tree. The application is called in the right-hand area of the screen.

The modeling pushbutton bar contains the following pushbuttons and provides the following navigation options:



- Previous object: Jumps to the application that was called before the present application.

  The navigation pane and tree display do not change, since these are displayed independently of the forwards/backwards navigation in the applications.

  Similarly, the open application is still displayed if you call another tree.

- Next object: Jumps to the application that was called after the present application.

  The navigation pane and tree display do not change, since these are displayed independently of the forwards/backwards navigation in the applications.

  Similarly, the open application is still displayed if you call another tree.

- Show/hide navigator: Hides the navigation pane and tree display if both are currently displayed.

  Shows the navigation pane if the tree display is shown.

  Shows the tree display if the navigation pane is shown.

  This function is only possible if an application has been called.

- Tree display full screen/half screen: Hides or shows the tree display.

  This is only possible if an application has been called.

You can remove the navigation pane and object tree from the display by choosing (hide navigation pane or hide tree).

You can display information on additional navigation functions in the navigation pane and information on the structure and functions of the object tree in the legends ( ) for the object trees.

3 Go and Create

For the Enterprise Data Warehousing scenario, we provide instructions for first development in the following areas:

Defining BI Objects

You can develop your own data models, which best represent the processes of your enterprise.

For information about modeling the first InfoProvider, see:

- Creating InfoObjects: Characteristic [Page 60] and Creating InfoObjects: Key Figures [Page 107]
- Creating DataStore Objects [Page 155]
- Creating InfoCubes [Page 130]

For information about creating the first transformation, see Creating Transformations [Page 242]

For information about creating the first data transfer process, see Creating Data Transfer Processes [Page 322]

For information about creating the first process chain, see Creating Process Chains [Page 402]

4 Core Development Tasks

This section forms the core of the Developer's Guides and describes the central areas of the development phase.

4.1 Developing Business Logic

Purpose

BI Content

BI Content is a collection of preconfigured role and task-based information models in the BI system. They are based on consistent metadata. BI Content provides selected roles within a company with the information they need to carry out their tasks.

For the Enterprise Data Warehousing area, these information models cover extractors, DataSources, process chains, InfoObjects, InfoSources, and InfoProviders.

You have to activate the BI Content objects before you can work with them.

For more information, see Installing Business Content [Page 435].

Modeling BI Objects

You can modify BI Content in line with your own requirements, or model completely new BI objects that fit your EDW scenario.

For more information about modeling data, see Modeling [Page 17] and the following sections:

Creating InfoObjects: Characteristics [Page 60]

Creating InfoObjects: Key Figures [Page 107]

Creating DataStore Objects [Page 155]

Creating InfoCubes [Page 130]

VirtualProviders [Page 165]

Creating InfoSets [Page 177]

Creating MultiProviders [Page 217]

For information about data transformations, see Transformations [Page 232] and Creating Transformations [Page 242].

For information about the data transfer process, see Data Transfer Process [Page 320] and Creating Data Transfer Processes [Page 322].

For information about the load process, see Scheduler [Page 336] and Creating InfoPackages [Page 337].

For information about process design, see Process Chains [Page 375] and Creating Process Chains [Page 402].

Modeling Non-Cumulatives

A non-cumulative is a non-aggregating key figure on the level of one or more objects that is always displayed in relation to time. Examples of non-cumulatives include headcount, account balance, and material inventory.

How you model the folder for non-cumulatives in the BI system depends on your scenario.

For more information, see Non-Cumulatives [Page 418].

Developing Customer or Partner Content

You can develop your own BI Content and deliver it to your business areas. You can also make limited modifications and changes to BI Content.

For more information about developing your own BI Content, see Customer and Partner Content [Page 444].

Implementing Routines

BI allows you to implement routines for transferring data. In this way, you reproduce your own business logic.

For more information, see Routines During the Transformation [Page 246].

In addition to the new transformation concept, you can still use the 3.x concept of transfer and update rules.

For more information, see

- Start Routines in Transfer Rules [Page 283] and Creating Transfer Routines [Page 283]
- Update Routines and Start Routines [Page 314] and Creating Update Routines [Page 317]

Prerequisites

Developing Customer or Partner Content:

Before you can develop customer and partner content, you may need to apply for your own namespace. This requires you to have an ABAP development license (ABAP/4 Development Workbench License).

4.1.1 Recommendations for Modeling Layers

Modeling a Data Warehouse Layer:

- If you are extracting data from heterogeneous source systems, we recommend that you model a data warehouse layer as an intermediate layer for consolidating the data. You use DataStore objects for this purpose.
- If you want to store data in the BI system at document level, we recommend that you use DataStore objects to build a data warehouse layer. The data that is stored there is only slightly denormalized and is stored at a useful level of granularity.
- The granularity of the data in the data warehouse layer should correspond to the granularity of the data as it was when obtained from the extractor.
- So that the information relevant for operational analysis is retained, the data is not aggregated in the data warehouse layer.
- The extractors for transaction and master data should be delta enabled.
- The data should contain the whole history.

Modeling the Architected Data Marts:

- Data in the architected data mart layer should meet data analysis requirements.

Modeling the Operational Data Store:

- Data in the operational data store should be stored with the highest possible level of granularity. This supports operational analysis.

General:

- Use the same central InfoObjects in all the layers. This is the only way to attain universal consistency.

You should avoid:

- Do **not** model the data model in such a way that you produce a *stove-pipe* data model. In this data model, the data flows straight into the architected data mart layer from the extractor.

4.1.2 Recommendations for Modeling the Data Flow

We recommend you model the data flow in accordance with a data warehouse layer architecture.

For information on a layer architecture, see Architecture of a Data Warehouse [Page 2].

You do not have to implement all four layers but can decide which structure best suits your needs.

Questions that are central to making this decision include:

- Does the data come from heterogeneous source systems and therefore need to be consolidated first?
- Do I need the data for operational data analysis?
- Which questions do I want to answer through analyzing this data?

For more information, see:

Simple Model Without Operational Data Store [Page 12]

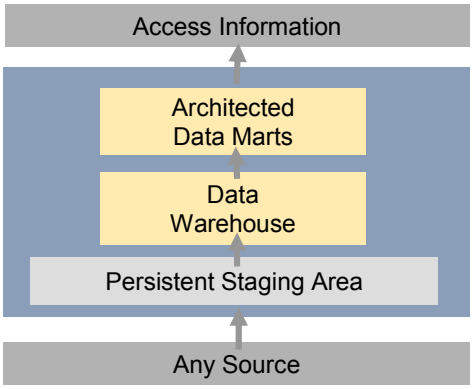
Modeling with All Layers [Page 13]

Modeling Without Persistent Data Storage [Page 14]

4.1.2.1 Simple Model Without Operational Data Store

Purpose

In this model data is loaded from homogeneous source systems or from one source system only and therefore does not require complex integration or consolidation processes:

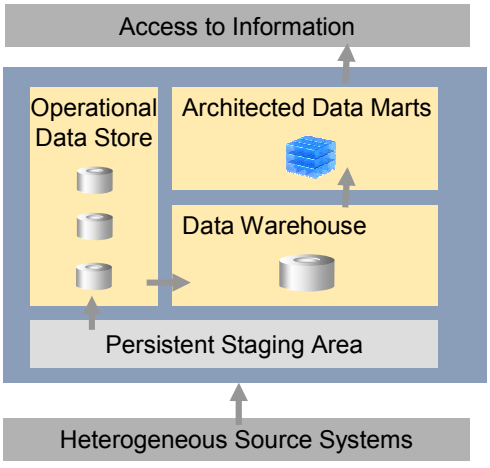


1. The data is saved in the data warehouse layer in Data Store objects. It is stored in granular form. The data structure is denormalized: Document headers and items are extracted using a DataSource and stored in the DataStore object.
2. The data is then written to an InfoCube in the architected data mart layer. The InfoCube contains data in an aggregated form. This is optimized for multidimensional analysis.

4.1.2.2 Modeling with All Layers

Purpose

In this model, a data warehouse is built including all the different layers. Data is loaded from heterogeneous source systems.



1. First, data is loaded into several DataStore objects in the operational data store layer and is saved there at document level.
2. The data is consolidated and updated to the data warehousing layer. It is saved there in one DataStore object. This DataStore objects allows flat reporting on all the data at a detailed level.

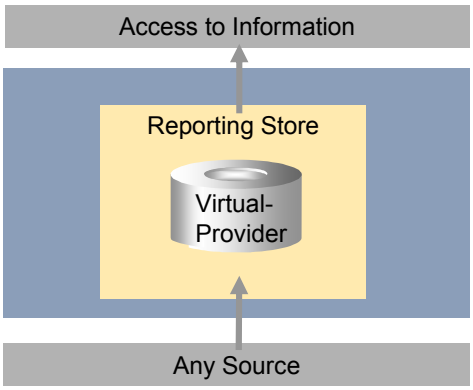
3. The data is updated to the architected data marts and saved there in an InfoCube in aggregated form. The InfoCube provides analysis-optimized access to the data and allows multidimensional analysis of aggregated data.

4.1.2.3 Modeling Without Persistent Data Storage

In this model, data is not stored persistently in the BI system but is read directly for reporting and analysis purposes.

Simple Model with VirtualProvider

The data is read straight from the source system using a VirtualProvider.



Analysis and reporting using a VirtualProvider is a suitable alternative if:

- you need up-to-date availability of data from the source system
- you only want to access small datasets sporadically, and replicating the data into BI would be too time-consuming in comparison
- only a few users execute queries simultaneously in the requested dataset.

As long as a suitable extractor is available, a query based on a VirtualProvider can replace drill through to an SAP source system using the report-report interface. In both cases, you obtain real-time data. However, a drill through uses an SAP query to present the data, whereas a VirtualProvider uses a BEx query.

Restrictions:

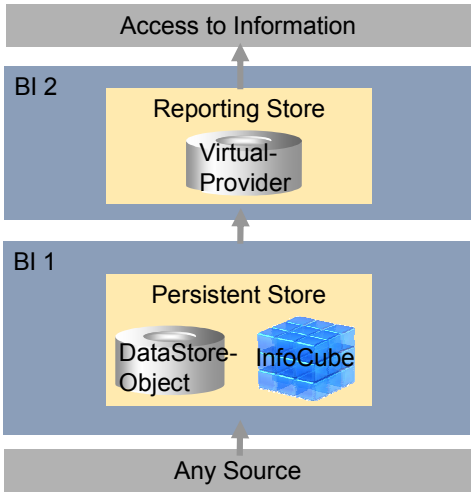
- The source system must always be available.
- Data analysis based on VirtualProviders represents additional system load on the source system. Performance is reduced further the more users are working with BI queries based on VirtualProviders and the higher the level of aggregation for the queries.
- VirtualProviders do not support navigation attributes.

You can also use VirtualProviders with a prototyping process with the generic extractor. You can also access non-SAP systems quickly and easily using DB Connect.

Since VirtualProviders do not require persistent data storage in the BI system, changes to the structures (extract structure, transfer structure, transfer rules) in the whole staging process of the prototype phase can be performed quickly.

Data Mart Scenario with VirtualProviders

First, data is stored in a BI system (BI 1) in DataStore objects or InfoCubes. The data is read for analysis purposes from an additional BI system (BI 2) using a VirtualProvider.



When you create a system landscape with several BI systems, you have two alternatives for distributing the transaction data: You can either replicate the data beforehand, or read it when it is requested from the source BI. With the second option, you use VirtualProviders. The restrictions that apply when you use VirtualProviders with a SAP system do not have the same significance when you join two BI systems. You can significantly reduce the performance load on the source system (BI 1) when using InfoCubes by implementing suitable aggregates or BI accelerator indexes. Limitations regarding system availability and navigation attributes still apply.

When you use a data mart scenario, it is not usually useful to replicate data at document level in all the connected BI systems. Since users only sporadically access the data in the connected BI systems in detail, this is a case that is well suited to the use of VirtualProviders.

4.1.3 Recommendations for Modeling InfoCubes

- Define lots of small dimensions rather than a few large dimensions.
- The size of the dimension tables should account for less than 10% of the fact table.
- If the size of the dimension table amounts to more than 10% of the fact table, mark the dimension as a line item dimension.
- If an InfoCube contains non-cumulative key figures, note the suggested performance tips [Page 434].

See also:

Differences Between Data Stores [Page 507]

4.1.4 Recommendations for Modeling InfoSets

- In order to avoid duplicate key figure values (see SAP Note 592785 on the SAP Service Marketplace), we recommend that you only make available the key figures for the query in the InfoSet for one DataStore object or InfoCube in the InfoSet. You do this in InfoSet maintenance by setting an indicator in the first column.
- We recommend that you only use one InfoSet object (DataStore object, InfoCube, or master data table) with ambiguous characteristic values. This means that when you join a DataStore object with an InfoCube, as long as the InfoCube contains the visible key figures, all the key characteristics of the DataStore object are used in the join condition for the InfoCube. Equally, when joining a master data table with compounding to an InfoCube, all of the key characteristics of the master data table are joined with the characteristics of the InfoCube.
- SAP does not generally support more than two InfoCubes in an InfoSet.
- As a general rule, do not use a large number of InfoProviders in an InfoSet. Instead, define more InfoSets.
- Do not use a large number of joins in an InfoSet, particularly if you are loading a high volume of data.
- Only use left outer joins in InfoSets when absolutely necessary. Performance is better for inner joins than left outer joins.
- Do not perform calculations before aggregation; this can produce incorrect query results.
- For time-dependent InfoSets, do not use the *Valid From* (0DATEFROM) and *Valid To* (0DATETO) fields in query definition; use the *Valid Time Interval* (VALIDTIMEINTERVAL) dimension. This is only visible in the Query Designer and serves to set time selections.

4.1.5 Recommendations for Modeling MultiProviders

- Model MultiProviders so that the characteristics you want to use in reporting and analysis are available in all the InfoProviders included in the MultiProvider.
- Use MultiProviders to keep the InfoProviders small.
- Ensure that the system only reads the relevant data when the query is executed by
  - using constants when you model the InfoProviders
  - modeling the InfoProviders with different key figures
  - including characteristic 0INFOPROV when you design the query for the MultiProvider so that you can filter by InfoProvider.
- Use MultiProviders to define a data analysis layer. This allows you to enhance analysis scenarios more simply. When you add an InfoProvider to a MultiProvider at a later point in time, the technical name of the query you have defined remains the same.
- Do not use more than one InfoCube with non-cumulative key figures in a MultiProvider since this may result in incorrect numbers in the query. Refer to SAP Note 690475 on SAP Support Portal.
- Do not perform calculations before aggregation; this can produce incorrect query results.

- Do not combine InfoProviders in a MultiProvider if they have very heterogeneous data models. If this is the case, use the report-report interface between queries that are defined on the basis of different InfoProviders.
- Do not use parts of compounded characteristics in InfoProviders. Refer to SAP Note 702542 on the SAP Support Portal.

Performance of MultiProviders

- Do not use more than 10 InfoProviders when you define a MultiProvider. If a number of InfoProviders are being read by a query at the same time, it can take a while for the system to calculate the results.
- If a MultiProvider contains an InfoCube with non-cumulative key figures, all subqueries are processed sequentially.
- You can only use OLAP tools for performance optimization (like caching and aggregation) for a MultiProvider if all the InfoProviders involved support this tool.

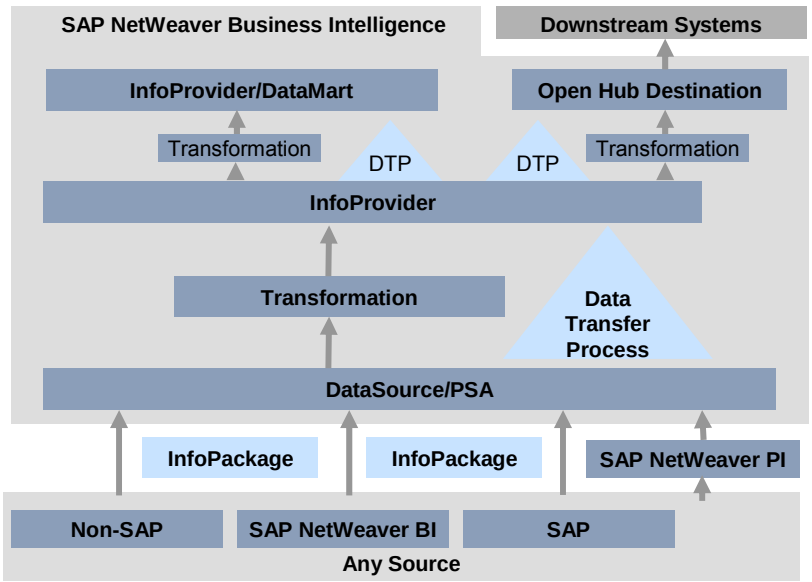
4.1.6 Modeling

Purpose

The tool you use for modeling is the Data Warehousing Workbench. Depending on your analysis and reporting requirements, different BI objects are available to you for integrating, transforming, consolidating, cleaning up, and storing data. BI objects allow efficient extraction of data for analysis and interpretation purposes.

Process Flow

The following figure outlines how BI objects are integrated into the dataflow:



Data that logically belongs together is stored in the source system as DataSources [Page 20]. DataSources are used for extracting data from a source system and transferring it into the BI system.

The Persistent Staging Area (PSA) [Page 46] in the BI system is the inbound storage area for data from the source systems. The requested data is saved, unchanged from the source system.

The transformation [Page 232] specifies how the data (key figures, time characteristics, characteristics) is updated and transformed from the source, into an InfoProvider [Page 125] or InfoSource. The transformation rules map the fields of the source to at least one InfoObject in the target. The information is mapped in structured form using the InfoObjects [Page 57].

You need to use an InfoSource [Page 261] if you want to execute two transformations one after the other.

Subsequently, the data can be updated to further InfoProviders [Page 125]. The InfoProvider provides the data that is evaluated in queries. You can also distribute data to other systems using the open hub destination [Page 651].

See also:

For more information about displaying the data flow for BI objects, see Data Flow Display [Page 19] section.

4.1.6.1 Namespaces for BI Objects

Use

The following namespaces [External] are generally available for BI objects:

SAP-delivery (Business Content) namespace:

- Objects beginning with 0
- Generated objects in the DDIC beginning with /BI0/

InfoCube 0SALES, fact table /BI0/FSALES

Customer namespace:

- Objects beginning with A-Z
- Generated objects in the DDIC beginning with /BIC/

InfoCube SALES, fact table /BIC/FSALES

Partner-specific namespace and customer-specific namespace:

- Object begins with /XYZ/ (example)

Special SAP namespaces for generated objects:

- The prefixes 1, 2, 3, 4, 6, 7, 8 are required in BW for DataSources and InfoSources in special SAP applications.
- The prefix 9A is required for the SAP APO application.

When you create your own objects, therefore, give them technical names that start with a letter. The maximum permitted length for a name varies from object to object. Typically, 9 to 11 letters can be used.

You can transfer from the Business Content version any Business Content objects that start with 0 and modify them to meet your requirements. If you change an InfoObject in the SAP namespace, your modified InfoObject is not overwritten immediately when you install a new release, and your changes remain in place for the time being.

You also have the option of enhancing the SAP Business Content. There is a partner namespace and a customer namespace available for you to do this. You have to request these namespaces specially. Once you have prepared the partner namespace and the customer namespace (for example, /XYZ/) you are able to create BI objects that start with the prefix /XYZ/. You use a forward slash (/) to avoid overlaps with SAP Business Content and customer-specific objects. For more information on this, see Using Namespace for Developing BW Objects [Page 450].

See also:

InfoObject Naming Conventions [Page 60]

Generating the Master Data Export DataSource [External]

Naming Conventions for Background Jobs [External]

4.1.6.2 Data Flow Display

Use

In the *Modeling* functional area of the Data Warehousing Workbench, you can use a graphic to display the dataflow of objects in BI. This illustrates the connections and dependencies between individual objects. You can call this graphic for InfoProviders and InfoSources.

Integration

From the object tree, you can call this graphic for an object by choosing *Display Dataflow* in the context menu (right mouse click) of the object. The graphic is displayed in the right-hand screen area of the Data Warehousing Workbench. All related objects and rules are displayed which precede this object in the data flow. For InfoSources with flexible update, all related objects are displayed up to and including the InfoProviders.

Features

The objects themselves represent nodes in the graphic. The rules that describe the dependencies between objects are displayed using arrows. An icon tells you what type of rule it is. The nodes contain the object symbol and the descriptive text. Which rule joins two objects and whether this is active or inactive is displayed using quick info if you select the corresponding arrow with the mouse. By choosing the appropriate pushbutton on the data flow graphic, you can display the technical name of the objects, print, enlarge, reduce, turn, refresh, and save the graphic in jpg form.

By double clicking on a node or arrow you branch to the display of an object or a rule.

4.1.6.3 DataSource

Definition

Set of fields that provide the data for a business unit for data transfer into BI. From a technical viewpoint, the DataSource is a set of logically-related fields that are provided to transfer data into BI in a flat structure (the extraction structure), or in multiple flat structures (for hierarchies).

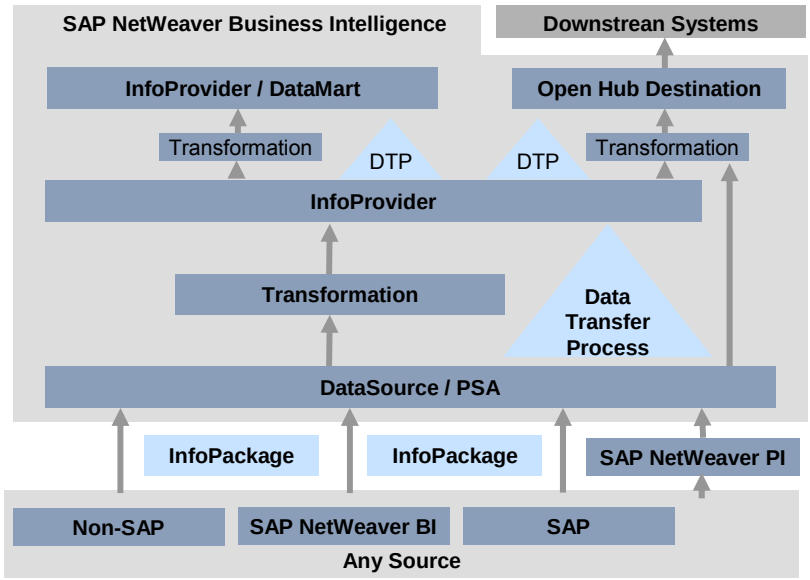
There are four types of DataSource:

- DataSource for transaction data
- DataSource for master data
  - DataSource for attributes
  - DataSource for texts
  - DataSource for hierarchies

Use

DataSources supply the metadata description of source data. They are used for extracting data from a source system and for transferring data into BI.

The following image illustrates the role of the DataSource in the BI data flow:



The data is loaded into BI from any source in the DataSource structure. An InfoPackage is used for this purpose. You determine the target into which data from the DataSource is to be updated during the transformation. You also assign DataSource fields to target object InfoObjects in BI.

Scope of DataSource Versus 3.x DataSource

3.x DataSource

DataSources were previously known as DataSource replicates (object type R3TR ISFS) in the BI system. The transfer of data from this type of DataSource (referred to as **3.x DataSources** below) is only possible if the 3.x DataSource is assigned to a 3.x InfoSource and the fields of the 3.x DataSource are assigned to 3.x InfoSource InfoObjects in transfer structure maintenance. A PSA table is generated when the 3.x transfer rules are activated, thus activating the 3.x transfer structure. Data can be loaded into this PSA table.

If your dataflow is modeled using objects that are based on the old concept (3.x InfoSource, 3.x transfer rules, 3.x update rules) and the process design is built on these objects, you can continue to work with 3.x DataSources when transferring data into BI from a source system.

DataSource

As of SAP NetWeaver 2004s, a new object concept is available for DataSources. It is used in conjunction with the changed objects concepts in data flow and process design (transformation, InfoPackage for loading to the PSA, data transfer process for data distribution within BI). The object type for a DataSource in the new concept - called **DataSource** in the following - is R3TR RSDS.

In DataSource maintenance in BI, you specify which DataSource fields contain the decision-relevant information for a business process and should therefore be transferred. When you activate the DataSource, the system generates a PSA table in the entry layer of BI. You can then load data into the PSA. You use an InfoPackage to specify the selection parameters for loading data into the PSA. In the transformation, you determine how the fields of the are assigned to the BI InfoObjects. Data transfer processes facilitate the further distribution of data from the PSA to other targets. The rules that you set in the transformation are applied here.

Overview of Object Types

A DataSource cannot exist simultaneously in both object types in the same system. The following table provides an overview of the (transport-relevant) metadata object types. The table also includes the object types for DataSources in SAP source systems:

DataSource Type	BI: Object Type of A or M Version	BI: Object Type of Shadow Version (Source System Independent)	SAP Source System: Object Type of A Version	SAP Source S Object Type o Version
DataSource	R3TR RSDS	R3TR SHDS (Shadow object delivered in its own table with release and version)	R3TR OSOD	R3TR OSOA
3.x DataSource	R3TR ISFS	R3TR SHFS for non-replicating source systems SHMP for replicating source systems, that is SAP source systems (shadow object delivered in its own	R3TR OSOD	R3TR OSOA

		table with source system key)		

Restrictions

The new DataSource concept cannot yet be used for transferring data from external systems (metadata and data transfer using staging BAPIs) or for transferring hierarchies.

Recommendation

We recommend that you adapt the data flow for the DataSource and the process design to the new concepts. If you want to migrate an existing data flow, first use an emulated 3.x DataSource and create a transformation for it. You can then migrate the 3.x DataSource to a DataSource and benefit from the new concepts in your scenario.

4.1.6.3.1 Functions for DataSources

Use

You can execute the following DataSource functions in the object tree of the Data Warehousing Workbench. The functions available differ depending on the object type (DataSource – RSDS, DataSource 3.x – ISFS) and source system:

- In the context menu of an application component, you can execute the following functions:
  - For both object types: Replicate metadata for all DataSources that are assigned to this application component.
  - For object type RSDS: Create DataSource.
- In the context menu of a DataSource, you can execute the following functions:
  - For both object types: Display, delete, manage, create transformation, create data transfer process, create InfoPackage.
  - For object type RSDS: Change, copy (however, not with an SAP source system as the target).
  - For object type ISFS: Create transfer rules, migrate.
  - Only for DataSources from SAP source systems (both object types): Display DataSource in source system, replicate metadata.

In the DataSource repository (transaction RSDS), you can execute the following functions. Here too, the functions available depend on the object type:

- For both object types: Display, delete, replicate.
- For object type RSDS: Change, create, copy (however, not with an SAP source system as the target), restore DataSource 3.x (if the DataSource is the result of migration and the migration was performed using the W*ith Export* option).
- For object type ISFS: Migrate.

Features

The following table provides an overview of the functions available in the Data Warehousing Workbench and DataSource repository for DataSources and DataSources 3.x:

Function	Description	More Information
Create	If you want to create a new DataSource for transferring data using UD Connect, DB Connect or from flat files, you first specify the name of the DataSource, the source system, where appropriate, and the data type of the DataSource. DataSource maintenance appears and you can enter the required data on the tab pages there.	DataSource Maintenance in BI [Page 25]
Display	The display mode of DataSource maintenance appears. You can display a DataSource 3.x or a DataSource (emulation). You cannot switch from the display mode to the change.	DataSource Maintenance in BI [Page 25] Emulation, Migration and Restoring DataSources [Page 37]
Change	The change mode of DataSource maintenance appears. For transfer of data from SAP source systems, you use this interface to select the fields from the DataSource to be transferred and to make specifications for format and conversion of field contents from the DataSource.	DataSource Maintenance in BI [Page 25]
Copy	You can use a DataSource as a template for creating a new DataSource. This function is not available if you want to use an SAP source system as the target. For SAP source systems, you can create DataSources in the source system in generic DataSource maintenance (RS02).	-
Delete	When you delete a DataSource, the dependent objects (such as a transformation or	-

	InfoPackage) are also deleted.	
Administration	The overview screen for requests in the PSA appears. Here you can select the requests that contain the data you want to call in PSA maintenance.	Persistent Staging Area [Page 42]
For SAP source systems: Display DataSource in source system	The DataSource display in the SAP source system appears.	-
For SAP source systems: Replicate metadata	The BI-relevant metadata for DataSources in SAP source systems is transferred into BI from the source system by means of replication.	Replicating DataSources [Page 525]
Create transformations	In the transformation, you determine how you want to assign the DataSource fields to InfoObjects in BI.	Creating Transformations [Page 242]
Create data transfer processes	In the data transfer process, you determine how you want to distribute the data from the PSA to additional targets in BI.	Creating Data Transfer Processes [Page 322]
Create InfoPackage	In the InfoPackage, you determine selections for transferring data into BI.	
For DataSources 3.x: Create transfer rules	If the DataSource 3.x is assigned to an InfoSource, determine how the DataSource fields are assigned to the InfoObjects of the InfoSource and how the data is to be transferred to the InfoObjects.	Processing Transfer Rules [Page 281]
For DataSources 3.x: Migrate	You can migrate a DataSource 3.x to a DataSource, that is, you can convert the metadata on the database. The DataSource 3.x can be restored to its status before the migration if the associated objects of DataSource 3.x (DataSource ISFS, mapping ISMP, transfer structure ISTS) are exported during migration. Before you perform migration, we recommend that you create the data flow with a transformation based on a DataSource 3.x. You also have the option of using an emulated DataSource 3.x.	Emulation, Migration and Restoring DataSources [Page 37]

4.1.6.3.2 DataSource Maintenance in BI

In DataSource maintenance in BI you can display DataSources and 3.x DataSources.

You can create or change DataSources for file source systems, UD Connect, DB Connect and Web services on this BI interface.

In DataSource maintenance, you can edit DataSources from SAP source systems. In particular, you can specify which fields you want to transfer into BI. In addition, you can determine properties for extracting data from the DataSource and properties for the DataSource fields. You can also change these properties.

You call DataSource maintenance from the context menu of a DataSource (*Display*, *Change*) or, if you are in the Data Warehousing Workbench, from the context menu of an application component in an object tree (*Create DataSource*). Alternatively you can call DataSource maintenance from the DataSource repository. In the Data Warehousing Workbench toolbar, choose *DataSource* to access the DataSource repository.

4.1.6.3.2.1 Editing DataSources from SAP Source Systems in BI

Use

A DataSource is defined in the SAP source system along with its properties and field list. In DataSource maintenance in BI, you determine which fields of the DataSource are to be transferred to BI. In addition, you can change the properties for extracting data from the DataSource and properties for the DataSource fields.

Prerequisites

You have replicated the DataSource in BI.

Procedure

You are in an object tree in the Data Warehousing Workbench.

1. Select the required DataSource and choose *Change.*
2. Go to the *General* tab page.

   Select *PSA Not Typed* if you do not want to generate the PSA for the DataSource in a typed structure but want to use character-type fields of type CHAR exclusively.

   Use this option if conversion during loading causes problems, for example, because there is no appropriate conversion routine, or if the source cannot guarantee that data is loaded with the correct data type.

   In this case, data can be loaded into the PSA after the DataSource has been activated and be corrected there.
3. Go to the *Extraction* tab page.

a. Under *Adapter*, you determine how the data is to be accessed. The options depend on whether the DataSource supports direct access and real-time data acquisition.

b. Select *Number Format* → *Direct Entry* to specify the character for the thousand separator and the decimal point character that you want to use for the DataSource fields. If a *User Master Record* has been specified, the system applies the settings of the user who is used when the conversion exit is executed. This is usually the BI background user (see also: User Management [External]).

4. Go to the *Fields* tab page.

   a. Under *Transfer*, specify the decision-relevant DataSource fields that should be available for extraction and transferred to BI.

   b. If required, change the setting for the *Format* of the field.

   c. If you choose an *External Format*, ensure that the output length of the field (external length) is correct. Change the entries, as required.

   d. If required, specify a conversion routine that converts data from an external format into an internal format.

   e. Under *Currency/Unit*, change the entries for the referenced currency and unit fields as required.

5. Check, save and activate your DataSource.

Result

When you activate the DataSource, BI generates a PSA table and a transfer program.

You can now create an InfoPackage. You define the selections for the data request in the InfoPackage. The data can be loaded into the entry layer of the BI system, the PSA. Alternatively you can access the data directly if the DataSource supports direct access and you have a VirtualProvider in the definition of the data flow.

4.1.6.3.2.2 Creating DataSources for File Source Systems

Use

Before you can transfer data from a file source system, the metadata (the file and field information) must be available in BI in the form of a DataSource.

Prerequisites

Note the following with regard to CSV files:

- Fields that are not filled in a CSV file are filled with a blank space if they are character fields and with a zero (0) if they are numerical fields.
- If separators are used inconsistently in a CSV file, the incorrect separator (which is not defined in the DataSource) is read as a character and both fields are merged into one field and may be shortened. Subsequent fields are no longer in the correct order.

Note the following with regard to CSV files and ASCII files:

- The conversion routine that are used determine whether you have to specify leading zeros. See Conversion Routines in BI Systems [Page 98].

- For dates, you usually use the format YYYYMMDD, without internal separators. Depending on the conversion routine that is used, you can also use other formats.

Notes on Loading

When you load external data, you can load the data into BI from any workstation. For performance reasons, however, you should store the data on an application server and load it into BI from there. This means that you can also load the data in the background.

If you want to load a large amount of transaction data into BI from a flat file and you can specify the file type of the flat file, you should create the flat file as an ASCII file. From a performance point of view, loading data from an ASCII file is the most cost-effective method. Loading from a CSV file takes longer because in this case, the separator characters and escape characters have to be sent and interpreted. In some circumstances, generating an ASCII file may involve more effort.

Procedure

You are in the Data Warehousing Workbench in the DataSource tree.

1. Select the application components in which you want to create the DataSource and choose *Create DataSource*.
2. On the next screen, enter a technical name for the DataSource, select the type of DataSource and choose *Copy*.

   The DataSource maintenance screen appears.
3. Go to the *General* tab page.
   a. Enter descriptions for the DataSource (short, medium, long).
   b. As required, specify whether the DataSource builds an initial non-cumulative and can return duplicate data records within a request.

      Specify whether you want to generate the PSA for the DataSource in untyped format: If the PSA is not typed it is not generated in a typed structure but is generated with character-like fields of type CHAR only.

      Use this option if conversion during loading causes problems, for example, because there is no appropriate conversion routine, or if the source cannot guarantee that data is loaded with the correct data type.

      In this case, after you have activated the DataSource you can load data into the PSA and correct it there.
4. Go to the *Extraction* tab page.
   a. Define the delta method for the DataSource.
   b. Specify whether you want the DataSource to support direct access to data.
   c. Real-time data acquisition is not supported for data transfer from files.
   d. Select the adapter for the data transfer. You can load text files or binary files from your local work station or from the application server.

      Text-type files only contain characters that can be displayed and read as text. CSV and ASCII files are examples of text files. For CSV files you have to specify a character that separates the individual field values. In BI you have to specify this separator character and an escape character which specifies this character as a component of the value, as required. After you have specified these characters you have to use them in the file. ASCII files contain data in a specified length. The defined field length in the file must be the same as the length of the assigned field in BI.

Binary files contain data in the form of Bytes. A file of this type can contain any type of Byte value, including Bytes that cannot be displayed or read as text. In this case, the field values in the file have to be the same as the internal format of the assigned field in BI.

Choose *Properties* if you want to display the general adapter properties.

e. Select the path to the file that you want to load or enter the name of the file directly, for example C:/Daten/US/Kosten97.csv.

You can also create a routine that determines the name of your file. If you do not create a routine to determine the name of the file, the system reads the file name directly from the *File Name* field.

f. Depending on the adapter and the file to be loaded, make further settings.

- **For binary files:**

  Specify the character record settings for the data that you want to transfer.

- **Text-type files:**

  Specify how many rows in your file are header rows and can therefore be ignored when the data is transferred.

  Specify the character record settings for the data that you want to transfer.

  For ASCII files:

  If you are loading data from an ASCII file, the data is requested with a fixed data record length.

  For CSV files:

  If you are loading data from an Excel CSV file, specify the *data separator* and the *escape character.*

  Specify the separator that your file uses to divide the fields in the *Data Separator* field.

  If the data separator character is a part of the value, the file indicates this by enclosing the value in particular start and end characters. You enter these start and end characters in the *Escape Characteristics* field.

  You chose the ; character as data separator. However, your file contains the value **12;45** for a field. If you set **“** as the escape character, the value in the file must be **12;45“** so that **12;45** is loaded into BI. The complete value that you want to transfer has to be enclosed by the escape characters.

  If the escape characters do not enclose the value but are used within the value, the system interprets the escape characters as a normal part of the value. If you have specified “ as the escape character, the value 12”45 is transferred as 12”45 and 12”45” is transferred as 12”45”.

  In a text editor (for example, note pad) check the escape character and separator currently being used in the file. These depend on the country version of the file you used.

Note that if you do not specify an escape character, the space character is interpreted as the escape character. We recommend that you use a different character as the escape character.

If you select the *Hex* indicator, you can specify the data separator and the escape character in hexadecimal format. When you enter a character for the data separator and the escape character, these are displayed as hexadecimal code after the entries have been checked. A two character entry for a data separator or an escape sign is always interpreted as a hexadecimal entry.

g. Make the settings for the number format (thousand separator and character used to represent a decimal point), as required.

h. Make the settings for currency conversion, as required.

i. Make any further settings that are dependent on your selection, as required.

5. Go to the *Proposal* tab page.

This tab page is only relevant for CSV files. For files in different formats, define the field list on the *Fields* tab page.

Here you create a proposal for the field list of the DataSource based on the sample data from your CSV file.

a. Specify the number of data records that you want to load and choose *Upload Sample Data*.

The data is displayed in the upper area of the tab page in the format of your file.

The system displays the proposal for the field list in the lower area of the tab page.

b. Choose *Copy to Field List* to select the fields that you want to transfer to the field list for the DataSource. All fields are selected by default.

6. Go to the *Fields* tab page.

Here you edit the fields that you transferred to the field list of the DataSource from the *Proposal* tab page. If you did not transfer the field list from a proposal, you can define the fields of the DataSource here.

a. To define a field, choose *Insert Row* and specify a field name.

b. Under *Transfer*, specify the decision-relevant DataSource fields that you want to be available for extraction and transferred to BI.

c. Instead of generating a proposal for the field list, you can enter InfoObjects to define the fields of the DataSource. Under *Template* InfoObject, specify InfoObjects for the fields in BI. This allows you to transfer the technical properties of the InfoObjects into the DataSource field.

Entering InfoObjects here does not equate to assigning them to DataSource fields. Assignments are made in the transformation. When you define the transformation, the system proposes the InfoObjects you entered here as InfoObjects that you might want to assign to a field.

d. Change the data type of the field as required.

e. Specify the key fields of the DataSource.

These fields are generated as a secondary index in the PSA. This is important in ensuring good performance for data transfer process selections, in particular with semantic grouping.

f. Specify whether lowercase is supported.

g. Specify whether the source provides the data in the internal or external format.

h. If you choose the external format, ensure that the output length of the field (external length) is correct. Change the entries, as required.

i. If required, specify a conversion routine that converts data from an external format into an internal format.

j. Select the fields for which you want to be able to set selection criteria when you schedule a data request using an InfoPackage. Data for this type of field is transferred in accordance with the selection criteria specified in the InfoPackage.

k. Choose the selection options (such as EQ, BT) that you want to be available for selection in the InfoPackage.

l. If required, in the field type, specify whether the data to be selected is language-dependent or time-dependent.

7. Check, save and activate the DataSource.

8. Go to the *Preview* tab page.

   If you select *Read Preview Data*, the number of data records you specified in your field selection is displayed in a preview.

   This function allows you to check whether the data formats and data are correct.

Result

The DataSource is created and is visible in the Data Warehousing Workbench in the DataSource overview for the file source system in the application component. When you activate the DataSource, the system generates a PSA table and a transfer program.

You can now create an InfoPackage. You define the selections for the data request in the InfoPackage. The data can be loaded into the inbound layer of the BI system, the PSA. Alternatively you can access the data directly if the DataSource supports direct access and you have defined a VirtualProvider in the data flow.

4.1.6.3.2.3 Creating DataSources for UD Connect

Use

To transfer data from UD Connect sources into BI, metadata (the information about the source object and source object elements) has to be available in BI in the form of a DataSource.

Prerequisites

You have connected a UD Connect source system.

Note the following background information:

Using InfoObjects with UD Connect [Page 583]

Data Types and Their Conversion [Page 583]

Using SAP Namespace for Generated Objects [Page 584]

Procedure

You are in the DataSource tree in the Data Warehousing Workbench.

1. Select the application components in which the DataSource is to be created and choose *Create DataSource*.
2. In the next screen, enter a technical name for the DataSource, select the type of the DataSource and choose *Copy*.

   The DataSource maintenance screen appears.
3. Go to the *General* tab page.
   a. Enter descriptions for the DataSource (short, medium, long).
   b. If necessary, specify if the DataSource builds an initial non-cumulative and supplies potentially duplicate data records within a request.
4. Go to the *Extraction* tab page.
   a. Define the delta method for the DataSource.
   b. Specify whether you want the DataSource to support direct access to data.
   c. UD Connect does not support real-time data acquisition.
   d. The system displays *Universal Data Connect (Binary Transfer)* as the adapter for the DataSource.

      Choose *Properties* if you want to display the general adapter properties.
   e. Select the UD Connect source object.

      The connection to the UD Connect source is established. All source objects that are available in the selected UD Connect source can be selected using input help. Source objects can be multi-dimensional storage or relational objects.
5. Go to the *Proposal* tab page.

   The system displays the elements of the source object (for JDBC for example fields, or for XMLA and ODBO characteristics and key figures) and creates a mapping proposal for DataSource fields. The mapping proposal is based on the similarity of the names of the source object element and DataSource field and the compatibility of the respective data types.

   Note that source object elements can have a maximum of 90 characters. Both upper and lower case are supported.
   a. Check the mapping and change the proposed mapping as required. Assign the non-assigned source object elements to free DataSource fields.

      You cannot map elements to fields if the types are incompatible. The system produces an error message in this case.
   b. Under *Copy to Field List*, select the fields that you want to transfer to the field list for the DataSource. All fields are selected by default.
6. Maintain the *Fields* tab page.

   Here you edit the fields that you transferred to the field list of the DataSource from the *Proposal* tab page.

a. Under *Transfer*, specify the decision-relevant DataSource fields that should be available for extraction and transferred to BI.

b. If required, change the values for the key fields of the source.

   These fields are generated as a secondary index in the PSA. This is important in ensuring good performance for data transfer process selections, in particular with semantic grouping.

c. If required, change the data type for a field.

d. Specify whether the source provides the data in the internal or external format.

e. If you choose an *External Format*, ensure that the output length of the field (external length) is correct. Change the entries, as required.

f. If required, specify a conversion routine that converts data from an external format into an internal format.

g. Select the fields for which you want to be able to set selection criteria when you schedule a data request using an InfoPackage. Data for this type of field is transferred in accordance with the selection criteria specified in the InfoPackage.

h. Choose the selection options (such as EQ, BT) that you want to be available for selection in the InfoPackage.

i. Under *Field Type*, specify whether the data to be selected is language-dependent or time-dependent, as required.

If you did not transfer the field list from a proposal, you can define the fields of the DataSource directly. Choose *Insert Row* and enter a field name. You can specify InfoObjects in order to define the DataSource fields. Under *Template InfoObject*, specify InfoObjects for the fields of the DataSource. This allows you to transfer the technical properties of the InfoObjects into the DataSource field.

Entering InfoObjects here does not equate to assigning them to DataSource fields. Assignments are made in the transformation. When you define the transformation, the system proposes the InfoObjects you entered here as InfoObjects that you might want to assign to a field.

7. Check, save and activate the DataSource.

8. Go to the *Preview* tab page.

   If you select *Read Preview Data*, the number of data records you specified in your field selection is displayed in a preview.

   This function allows you to check whether the data formats and data are correct.

Result

The DataSource is created and added to the DataSource overview for the UD Connect source system in the application component in the Data Warehousing Workbench. When you activate the DataSource, the system generates a PSA table and a transfer program.

You can now create an InfoPackage. You define the selections for the data request in the InfoPackage. The data can be loaded into the entry layer of the BI system, the PSA. Alternatively you can access the data directly if the DataSource allows direct access and you have a VirtualProvider in the definition of the data flow.

4.1.6.3.2.4 Creating DataSources for DB Connect

Use

Before you can transfer data from a database source system, the metadata (the table, view and field information) must be available in BI in the form of a DataSource.

Prerequisites

See Requirements for Database Tables or Views [Page 609]

You have connected a DB Connect source system.

Procedure

You are in the Data Warehousing Workbench in the DataSource tree.

1. Select the application components in which you want to create the DataSource and choose *Create DataSource*.
2. On the next screen, enter a technical name for the DataSource, select the type of DataSource and choose *Copy*.

   The DataSource maintenance screen appears.
3. Go to the *General* tab page.
   a. Enter descriptions for the DataSource (short, medium, long).
   b. As required, specify whether the DataSource builds an initial non-cumulative and can return duplicate data records within a request.
4. Go to the *Extraction* tab page.
   a. Define the delta process for the DataSource.
   b. Specify whether you want the DataSource to support direct access to data.
   c. The system displays *Database Table* as the adapter for the DataSource.

      Choose *Properties* if you want to display the general adapter properties.
   d. Select the source from which you want to transfer data.
      - Application data is assigned to a database user in the Database Management System (DBMS). You can specify a database user here. In this way you can select a table or view that is in the schema of this database user. To perform an extraction, the database user used for the connection to BI (also called BI user) needs read permission in the schema of the database user.

        If you do not specify the database user, the tables and views of the BI user are offered for selection.
      - Call the value help for field *Table/View*.

        In the next screen, select whether tables and/or views should be displayed for selection and enter the necessary data for the selection under *Table/View*. Choose *Execute*.

- The database connection is established and the database tables are read. The *Select DB Object Names* screen appears. The tables and views belonging to the specified database user that correspond to your selections are displayed on this screen. The technical name, type and database schema for a table or view are displayed.

  Only use tables and views in the extraction whose technical names consist solely of upper case letters, numbers, and underscores (_). Problems may arise if you use other characters.

  Extraction and preview are only possible if the database user used in the connection (BI user) has read permission for the selected table or view.

  Some of the tables and views belonging to a database user might not lie in the schema of the user. If the responsible database user for the selected table or view does not match the schema, you cannot extract any data or call up a preview. In this case, make sure that the extraction is possible by using a suitable view. For more information, see Database Users and Database Schemas [Page 610].

5. Go to the *Proposal* tab page.

   The fields of the table or view are displayed here. The overview of the database fields tells you which fields are key fields, the length of the field in the database compared with the length of the field in the ABAP data dictionary, and the field type in the database and the field type in the ABAP dictionary. It also gives you additional information to help you check the consistency of your data.

   A proposal for creating the DataSource field list is also created. Based on the field properties in the database, a field name and properties are proposed for the DataSource. Conversions such as from lowercase to uppercase or from " " (space) to "_" (underlining) are carried out. You can also change names and other properties of the DataSource field. Type changes are necessary, for example, if a suitable data type is not proposed. Changes to the name could be necessary if the first 16 places of field names on the database are identical. The field name in the DataSource is truncated after 16 places, so that a field name could occur more than once in proposals for the DataSource.

   When you use data types, be aware of database-specific features. For more information, see Requirements for Database Tables and Views [Page 609].

6. Choose *Copy to Field List* to select the fields that you want to transfer to the field list for the DataSource. All fields are selected by default.

7. Go to the *Fields* tab page.

   Here you edit the fields that you transferred to the field list of the DataSource from the *Proposal* tab page.

   a. Under *Transfer*, specify the decision-relevant DataSource fields that should be available for extraction and transferred to BI.

   b. If required, change the values for the key fields of the source.

      These fields are generated as a secondary index in the PSA. This is important in ensuring good performance for data transfer process selections, in particular with semantic grouping.

   c. Specify whether the source provides the data in the internal or external format.

d. If you choose an *External Format*, ensure that the output length of the field (external length) is correct. Change the entries, as required.

e. If required, specify a conversion routine that converts data from an external format into an internal format.

f. Select the fields for which you want to be able to set selection criteria when you schedule a data request using an InfoPackage. Data for this type of field is transferred in accordance with the selection criteria specified in the InfoPackage.

g. Choose the selection options (such as EQ, BT) that you want to be available for selection in the InfoPackage.

h. Under *Field Type*, specify whether the data to be selected is language-dependent or time-dependent, as required.

8. Check the DataSource.

   The field names are checked for upper and lower case letters, special characters, and field length. The system also checks whether an assignment to an ABAP data type is available for the fields.

9. Save and activate the DataSource.

10. Go to the *Preview* tab page.

    If you choose *Read Preview Data*, the specified number of data records, corresponding to your field selection, is displayed in a preview.

    This function allows you to check whether the data formats and data are correct. If you can see in the preview that the data is incorrect, try to localize the error.

    See also: Localizing Errors [Page 618]

Result

The DataSource is created and is visible in the Data Warehousing Workbench in the DataSource overview for the database source system under the application component. When you activate the DataSource, the system generates a PSA table and a transfer program.

You can now create an InfoPackage. You define the selections for the data request in the InfoPackage. The data can be loaded into the entry layer of the BI system, the PSA. Alternatively you can access the data directly if the DataSource supports direct access and you have a VirtualProvider in the definition of the data flow.

4.1.6.3.2.5 Creating DataSources for Web Services

Use

In order to transfer data into BI using a Web service, the metadata first has to be available in BI in the form of a DataSource.

Procedure

You are in the DataSource tree in the Data Warehousing Workbench.

1. Select the application components in which the DataSource is to be created and choose *Create DataSource*.

2. In the next screen, enter a technical name for the DataSource, select the type of the DataSource and choose *Copy*.

   The DataSource maintenance screen appears.

3. Go to the *General* tab page.

   a. Enter descriptions for the DataSource (short, medium, long).

   b. If necessary, specify whether the DataSource may potentially deliver duplicate data records within a request.

4. Go to the *Extraction* tab page.

   Define the delta method for the DataSource.

   DataSources for Web services support real-time data acquisition. Direct access to data is not supported.

5. Go to the *Fields* tab page.

   Here you determine the structure of the DataSource either by defining the fields and field properties directly, or by selecting an InfoObject as a *Template InfoObject* and transferring its technical properties for the field in the DataSource. You can modify the properties that you have transferred from the InfoObject further to suit your requirements by changing the entries in the field list.

   Entering InfoObjects here does not equate to assigning them to DataSource fields. Assignments are made in the transformation. When you define the transformation, the system proposes the InfoObjects you entered here as InfoObjects that you might want to assign to a field.

6. Save and activate the DataSource.

7. Go to the *Extraction* tab page.

   The system has generated a function module and a Web service with the DataSource. They are displayed on the *Extraction* tab page. The Web service is released for the SOAP runtime.

8. Copy the technical name of the Web service and choose *Web Service Administration*.

   The administration screen for SOAP runtime appears. You can use the search function to find the Web service. The Web service is displayed in the tree of the *SOAP Application for RFC-Compliant FMs*. Select the Web service and choose *Web Service* → *WSDL* (Web Service Description Language) to display the WSDL description.

Result

The DataSource is created and is visible in the Data Warehousing Workbench in the application component in the DataSource overview for the Web service source system. When you activate the DataSource, the system generates a PSA table and a transfer program.

Before you can use a Web service to transfer data into BI for the DataSource, create a corresponding InfoPackage (push package). If an InfoPackage is already available for the DataSource, you can test the Web service push in Web service administration.

See also:

Web Services [External]

4.1.6.3.3 Emulation, Migration and Restoring DataSources

Emulation

3.x DataSources (object type R3TR ISFS) exist in the BI database in the metadata tables that were available in releases prior to SAP NetWeaver 2004s.

The emulation permits you to display and use the DataSource 3.x using the interfaces of the new DataSource concept. The DataSource (R3TR RSDS) is instantiated from the metadata tables of the DataSource 3.x.

You can display a 3.x DataSource as an emulated DataSource in DataSource maintenance in BI. You can also model the data flow with transformations for an emulated DataSource if there are already active transfer rules and a transfer structure and a PSA for the 3.x DataSource. Once you have defined the objects of the data flow, you can set the processes for data transfer (loading process using InfoPackage and data transfer process), along with other data processing processes in BI. We recommend that you use process chains.

Emulation and definition of the objects and processes of the data flow that are based on the emulation in accordance with the new concept are a preparatory step in migrating the DataSource.

If you use an emulated DataSource 3.x, note that the InfoPackage does not use all of the settings defined in the 3.x data flow because in the new data flow it only loads the data into the PSA. To prevent problems arising from misunderstandings about using the InfoPackage, we recommend that you only use the emulation in development and test systems.

More Information:

Using Emulated 3.x DataSources [Page 626]

Migration

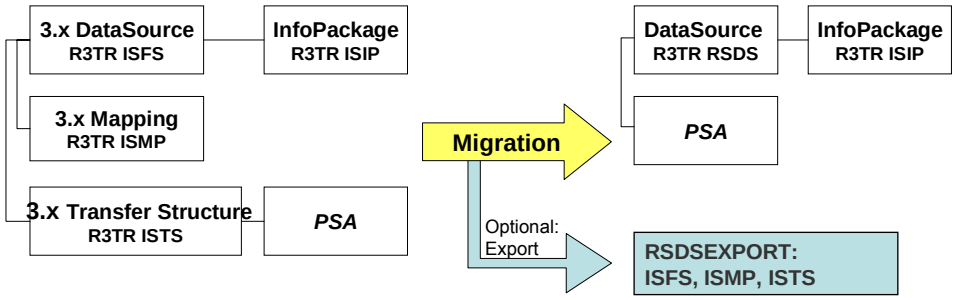
You can migrate a 3.x DataSource that transfers data into BI from an SAP source system or a file or uses DB Connect to transfer data into a DataSource. 3.x XML DataSources and 3.x DataSources that use UD Connect to transfer data cannot be migrated directly. However, you can use the 3.x versions as a copy template for a Web service or UD Connect DataSource.

You cannot migrate hierarchy DataSources, DataSources that use the IDoc transfer method, export DataSources (namespace 8* or /*/8*) or DataSources from BAPI source systems.

Migration (SAP Source Systems, File, DB Connect)

If the 3.x DataSource already exists in a data flow based on the old concept, you use emulation first to model the data flow with transformations and data transfer processes and then test it. During migration you can delete the data flow you were using before, along with the metadata objects.

If you are using real-time data acquisition or want to access data directly using the data transfer process, we recommend migration. Emulation does not support this.



When you migrate a 3.x DataSource (R3TR ISFS) in an original system, the system generates a DataSource (R3TR RSDS) with a transport connection. The 3.x DataSource is deleted, along with the 3.x metadata object mapping (R3TR ISMP) and transfer structure (R3TR ISTS), which are dependent on it. If a PSA and InfoPackages (R3TR ISIP) already exist for the 3.x DataSource, they are transferred to the migrated DataSource, along with the requests that have already been loaded. After migration, only the specifications about how data is loaded into the PSA are used in the InfoPackage.

You can export the 3.x objects, 3.x DataSource, mapping and transfer structure during the migration so that these objects can be restored. The collected and serialized objects are stored in a local table (RSDSEXPORT).

You can now transport the migration into the target system.

When you import the transport into the target system in the after import, the system migrates the 3.x DataSource (R3TR ISFS) (as long as it is available in the target system) to a local DataSource (R3TR RSDS), without exporting the objects that are to be deleted. The 3.x DataSource, mapping (R3TR ISMP) and transfer structure (R3TR ISTS) objects are deleted and the related InfoPackages are migrated. The data in the DataSource (R3TR RSDS) is transferred to the PSA.

More Information:

Migrating 3.x DataSources [Page 40]

Migrating by Copying

You cannot migrate in the way described above

- If you are transferring data into BI using a Web service and have previously used XML DataSources that were created on the basis of a file DataSource.
- If you are transferring data into BI using UD Connect and have previously used a UD Connect DataSource that was generated using an InfoSource.

3.x XML DataSource → Web Service DataSource

You can make a copy of a generated 3.x XML DataSource in a source system of type *Web Service*. When you activate the DataSource, the system generates a function module and a Web service. On your interface, these are different to the 3.x objects. The 3.x objects (3.x DataSource, mapping, transfer rules and generated function module and Web service) are therefore obsolete and can be deleted manually.

3.x UD Connect DataSource → UD Connect DataSource

For a 3.x UD Connect DataSource, you can make a copy in a source system of type *UD Connect*. The 3.x objects (3.x DataSources, mapping, transfer rules and the generated function module) are obsolete after they have been copied and can be deleted manually.

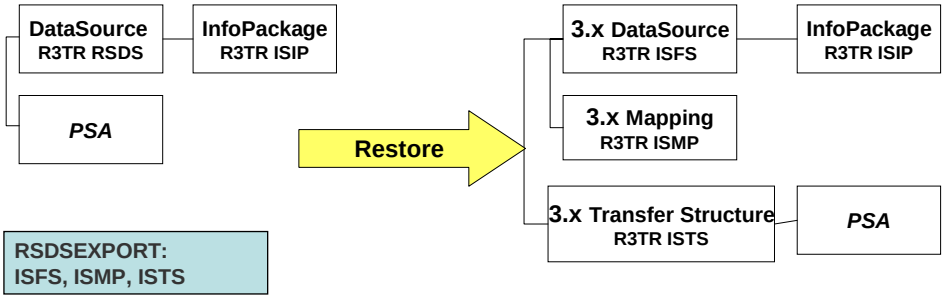
More Information:

Migrating 3.x DataSources (UD Connect, Web Service) [Page 41]

Restoring

For SAP source systems, files and DB Connect, you can restore a 3.x DataSource from a DataSource (R3TR RSDS) as long as you exported the 3.x metadata objects when the 3.x DataSource was migrated in the original system and in so doing, archived them. The system reproduces the 3.x DataSource (R3TR ISFS), mapping (R3TR ISMP), and transfer structure (R3TR ISTS) objects with their pre-migration status.

Only use this function if unexpected problems occur with the new data flow after migration and these problems can only be solved by restoring the data flow used previously.



When you restore, the 3.x DataSource (R3TR ISFS), mapping (R3TR ISMP) and transfer structure (R3TR ISTS) objects that were exported are generated with a transport connection in the original system. The DataSource (R3TR RSDS) is deleted. The system tries to retain the PSA. This is only possible if a PSA existed for the 3.x DataSource before migration. This may not be the case if an active transfer structure did not exist for the 3.x DataSource or if the data for the DataSource was loaded using an IDoc. The InfoPackage (R3TR ISIP) for the DataSource is retained in the system. Available targets are displayed in the InfoPackage (this also applies to InfoPackages that were created after migration). However, in InfoPackage maintenance, you have to reselect the targets into which you want to update data.

The transformation (R3TR TRFN) and data transfer process (R3TR DTPA) objects that are dependent on the DataSource (R3TR RSDS) are retained and can be deleted manually, as required. You can no longer use data transfer processes for direct access or real-time data acquisition.

You can now transport the restored 3.x DataSource and the dependent transfer structure and mapping objects into the target system.

When you transport the restored 3.x DataSource into the target system, the DataSource (R3TR RSDS) is deleted in the after image. The PSA and InfoPackages are retained. If a transfer structure (R3TR ISTS) is transported with the restore process, the system tries to transfer the PSA for this transfer structure. This is not possible if no transfer structure exists when you restore the 3.x DataSource or if IDoc is specified as the transfer method for the 3.x DataSource. The PSA is retained in the target system but is not assigned to a DataSource/3.x DataSource or to a transfer structure.

4.1.6.3.3.1 Using Emulated 3.x DataSources

Use

You can display an emulated 3.x DataSource in DataSource maintenance in BI. Changes are not possible in this display. In addition, you can use emulation to create the (new) data flow for a 3.x DataSource with transformations, without having to migrate the existing data flow that is based on the 3.x DataSource.

We recommend that you use emulation before migrating the DataSource in order to model and test the functionality of the data flow with transformations, without changing or deleting the objects of the existing data flow. Note that use of the emulated Data Source in a data flow with transformations has an effect on the evaluation of the settings in the InfoPackage. We therefore recommend that you only use the emulation in a development or test system.

Constraints

An emulated 3.x DataSource does not support real-time data acquisition, using the data transfer process to access data directly, or loading data directly (without using the PSA).

Prerequisites

If you want to use transformations in the modeling of the data flow for the 3.x DataSource, the transfer rules and therefore the transfer structure must be activated for the 3.x DataSource. The PSA table to which the data is written is created when the transfer structure is activated.

Procedure

To display the emulated 3.x DataSource in DataSource maintenance, highlight the 3.x DataSource in the DataSource tree and choose *Display* from the context menu.

To create a data flow using transformations, highlight the 3.x DataSource in the DataSource tree and choose *Create Transformation* from the context menu. You also use the transformation to set the target of the data transferred from the PSA.

To permit a data transfer to the PSA and further updating of the data from the PSA to the InfoProvider, select the DataSource 3.x in the DataSource tree and choose *Create InfoPackage* or *Create Data Transfer Process* in the context menu. We recommend that you use the processes for data transfer to prepare for the migration of a data flow and not in the production system.

Result

If you defined and tested the data flow with transformations using the emulation, you can migrate the DataSource 3.x after a successful test.

4.1.6.3.3.2 Migrating 3.x DataSources

Use

To take advantage of the new concepts in a data flow using 3.x objects, you must migrate the data flow and the 3.x objects it contains.

Procedure

1. In the original system (development system), in the Data Warehousing Workbench, choose *Migrate* in the context menu of the 3.x DataSource.
2. If you want to restore the 3.x DataSource at a later time, choose *With Export* on the next screen.
3. Specify a transport request.
4. Transport the migrated DataSource to the target system (quality system, productive system).
5. Activate the DataSource in the target system.

4.1.6.3.3.3 Migrating 3.x DataSources (UD Connect, Web Service)

Use

To take advantage of the new concepts in a data flow using 3.x objects, you must migrate the data flow and the 3.x objects it uses. 3.x XML DataSources and 3.x UD Connect DataSources cannot be migrated in the standard way because the 3.x objects are created in the Myself system and in the new data flow the DataSources need to be created in separate source systems for Web Service and UD Connect. However, you can nevertheless “migrate“ a 3.x DataSource of this type. This involves copying the 3.x DataSource into a source system.

Prerequisites

The UD Connect source system and the Web service source system are available.

The UD Connect source system uses the same RFC destination, and therefore the same BI Java Connector, as the 3.x DataSource.

Procedure

1. In the original system (development system), in the Data Warehousing Workbench, choose *Copy* in the context menu of the 3.x DataSource.
2. On the next screen, enter the name of the DataSource under *DataSource*.
3. Under *Source System*, specify the Web service or UD Connect source system to which you want to migrate the DataSource.
4. Delete the dependent 3.x objects (3.x DataSource, mapping, transfer rules and any generated function modules and the Web service).
5. Transport the DataSource and the deletion of 3.x objects into the target system.
6. Activate the DataSource.

Result

When you activate the Web service DataSource, the system generates a Web service and an rfc-compliant function module for the data transfer.

When you activate the UD Connect DataSource, the system generates a function module for extraction and data transfer.

4.1.6.3.3.4 Restoring 3.x DataSources

Use

In the original system, you can restore 3.x DataSources from DataSources that were migrated in the standard way (SAP source system, file, DB Connect). You use a transport to restore the 3.x DataSource in the target system.

Only use this function if unexpected problems occur with the new data flow after migration and these problems can only be solved by restoring the data flow used previously.

Prerequisites

You exported and archived the relevant 3.x objects when you migrated the 3.x DataSource.

Procedure

1. In the original system (development system), in the maintenance for the migrated DataSource (transaction RSDS), choose **DataSource** → **Restore**.
2. Specify a transport request.
3. If required, delete the dependent transformation (R3TR TRFN) and data transfer process (R3TR DTPA) objects.
4. Transport the restored 3.x DataSource (R3TR ISFS), along with its dependent objects, into the target system.

4.1.6.4 Persistent Staging Area

Purpose

The Persistent Staging Area (PSA) is the inbound storage area for data from the source systems in the BI system. The requested data is saved, unchanged from the source system. Request data is stored in the transfer structure format in transparent, relational database tables of the BI system in which the request data is stored in the format of the DataSource. The data format remains unchanged, meaning that no summarization or transformations take place, as is the case with InfoCubes.

When loading flat files, the data does not remain completely unchanged, since it is adjusted by conversion routines if necessary (for example, the date format 31.21.1999 is converted to 19991231 in order to ensure uniformity of data).

The possible coupling of the load process from the further processing in BI contributes to an improved load performance. The operative system is not debited if data errors first appear with further processing.

The PSA delivers the backup status for the ODS layer (until the total staging process is confirmed). The duration of the data storage in the PSA is medium-term, since the data can still be used for reorganization. However, for updates to DataStore objects, data is stored only for the short-term.

Features

A transparent PSA table is created for every DataSource that is activated. The PSA tables each have the same structure as their respective DataSource. They are also flagged with key fields for the request ID, the data package number, and the data record number.

InfoPackages load the data from the source into the PSA. The data from the PSA is processed with data transfer processes.

With the context menu entry *Manage* for a DataSource in the Data Warehousing Workbench you can go to the PSA maintenance for data records of a request or delete request data from the PSA table of this DataSource. You can also go to the PSA maintenance from the monitor for requests of the load process.

Using partitioning, you can separate the dataset of a PSA table into several smaller, physically independent, and redundancy-free units. This separation can mean improved performance when you update data from the PSA. In the Implementation Guide with *SAP NetWeaver* → *Business Intelligence* → *Connections to Other Systems* → *Maintain Control Parameters for Data Transfer* you define the number of data records needed to create a new partition. Only data records from a complete request are stored in a partition. The specified value is a threshold value.

Constraints

The number of fields is limited to a maximum of 255 when using TRFCs to transfer data. The length of the data record is limited to 1962 bytes when you use TRFCs.

4.1.6.4.1 DB Storage Parameters

Use

You can maintain database storage parameters for PSA tables, master data tables, InfoCube fact- and dimension tables, and DataStore object tables.

Use this setting to determine how the system handles the table when it creates it in the database:

- Use *Data Type* to set in which physical database area (tablespace) the system is to create the table. Each data type (master data, transaction data, organization- and Customizing data, customer data) has its own physical database area, in which all tables assigned to this data type are stored. If selected correctly, your table is automatically assigned to the correct area when it is created in the database.

  We recommend you use separate tablespaces for very large tables.

  You can find information about creating a new data type in SAP Note 0046272 (Introduce new data type in technical settings).

- Via *Size Category*, you can set the amount of space the table is thought to need in the database. Five categories are available in the input help. You can also see here how many data records correspond to each individual category. When creating the table, the system reserves an initial storage space in the database. If the table later requires more storage space, it obtains it as set out in the size category. Correctly setting the size category prevents there being too many small extents (save areas)

for a table. It also prevents the wastage of storage space when creating extents that are too large.

You can use the maintenance for storage parameters to better manage databases that support this concept.

You can find additional information about the data type and size category parameters in the ABAP Dictionary table documentation, under Technical Settings [External].

PSA Table

For PSA tables, you access the database storage parameter maintenance by choosing *Goto → Technical Attributes* in DataSource maintenance. In dataflow 3.x, you access this setting in transfer rule maintenance in the *Extras* menu.

You can also assign storage parameters for a PSA table already in the system. However, this has no effect on the existing table. If the system generates a new PSA version (a new PSA table) due to changes to the DataSource, this is created in the data area for the current storage parameters.

InfoObject Tables

For InfoObject tables, the maintenance of database storage parameters can be found in the InfoObject maintenance, in the *Extras* menu.

InfoCube/aggregate fact and dimension tables

For fact- and dimension tables, the maintenance of database storage parameters can be found in the InfoCube maintenance, in the *Extras* menu.

DataStore Object Tables (Activation Queue and Table for Active Data)

For DataStore object tables, the maintenance of database storage parameters can be found in DataStore object maintenance, in the *Extras* menu.

4.1.6.4.2 Deleting Requests from the PSA

Use

With this function you delete requests from the PSA. This reduces the volume of data in the PSA.

Applications are, for example, deleting incorrect requests or deleting delta requests that were updated successfully in an InfoProvider and for which no further deltas should be loaded.

Prerequisites

You checked that the PSA data that you want to delete is no longer needed.

Procedure

Including the deletion of requests from the PSA in process chains

You are in the plan view of the process chain in which you want to insert the process variant.

1. To insert a process variant for deleting requests from the PSA in the process chain, select process type *Deletion of Requests from the PSA* from process category *Further BI Processes* by double-clicking.
2. In the next dialog box, enter a name for the process variant and choose *Create*.
3. In the next screen, enter a description for the process variant and choose *Continue*.

   The maintenance screen for the process variant appears.

   Here you define the DataSource for which requests should be deleted from the PSA and the relevant selections.
4. Under *Type of Object*, determine the object type that you want to use to select the requests that are to be deleted.

   You can make the selection either directly by specifying a PSA table or implicitly by specifying an InfoPackage or a data transfer process.
5. Under *Object Name* select one or more PSA tables, InfoPackages or data transfer processes.

   InfoPackages and data transfer processes provide the information about the DataSource to which the PSA requests should be deleted. Data transfer processes are therefore only offered if their source is a DataSource.

   During deletion, all the PSA requests of the DataSource for which the specified InfoPackage was created are taken into consideration – not only the PSA requests of the DataSource loaded with this InfoPackage.

   Similarly, all the PSA requests of the DataSource that are the source fort he DTP are taken into consideration during deletion – not only the PSA requests already updated from the PSA with this DTP (source requests).
6. Define the deletion conditions for the requests of the selected PSA tables.

   You can define how old the requests may be before they are deleted. You can also define if the request status should be taken into consideration in the selection of the requests to be deleted.
7. Save your entries and return to the previous screen.
8. On the following screen, confirm the addition of the process variants in the process chain.

   You arrive in the plan view for the process chain. The process variant for deleting requests from the PSA is included in your process chain.

Deleting requests for a DataSource in the Data Warehousing Workbench from the PSA

You are in an object tree in the Data Warehousing Workbench.

1. Select the DataSource for which you want to delete requests from the PSA and choose *Manage*.
2. On the next screen, select one or more requests from the list and choose *Delete Request from DB*.
3. When asked whether you want to delete the request(s), confirm.

   The system deletes the requests from the PSA table.

You can also delete requests in DataSource maintenance. Choose *Goto* → *Manage PSA* (pushbutton). Starting with step 2, proceed as described above.

Note

The change log is stored as a PSA table. For information about deleting requests from the change log, see Deleting from the Change Log [External].

4.1.6.4.3 Previous Technology of the PSA

The PSA is the entry layer for data in BI. The data is updated to PSA tables that were generated for active DataSources during the load process. The PSA is managed with a DataSource.

The previous technology of the PSA was oriented to the transfer structure. The PSA table is generated for an active transfer structure in this case. The PSA as a standalone application is managed in an object tree of the Administrator Workbench.

You can still use this technology when your data model is based on the previously available objects and rules (DataSource 3.x, transfer rule 3.x, update rule 3.x). However, we recommend that you use the concepts for DataSources and transformations available after SAP NetWeaver 2004s, which includes using the new technology of the PSA.

4.1.6.4.3.1 ersistent Staging Area

Purpose

The Persistent Staging Area (PSA) is the inbound storage area in BI for data from the source systems. The requested data is saved, unchanged from the source system.

Request data is stored in the transfer structure format in transparent, relational database tables in BI. The data format remains unchanged, meaning that no summarization or transformations take place, as is the case with InfoCubes.

When loading flat files, the data does not remain completely unchanged, since it is adjusted by conversion routines, where necessary (for example, the date format 31.21.1999 is converted to 19991231 in order to ensure uniformity of data).

You determine the *PSA* transfer method in transfer rule maintenance [Page 281].

If you set the PSA when you are extracting data, you get improved performance if you use TRFCs for loading the data. The temporary storage facility in the PSA also allows you to check and change the data before the update into data targets. Coupling the load process for further processing in BI also contributes to an improved load performance. In contrast to a data request with IDocs, a data request in the PSA also gives you various options for further updating [Page 47] data to the data targets. Coupling the load process for further processing in BI also contributes to an improved loading performance. If errors occur when data is processed further, the operative system is not affected.

The PSA delivers the backup status for the ODS (until the total staging process is confirmed). The duration of the data storage in the PSA is medium-term, since the data can still be used for reorganization. However, for updates to ODS objects, data is stored only for the short-term.

In the PSA tree of the Administrator Workbench, a PSA is displayed for every InfoSource. You get to the PSA tree in the Administrator Workbench using either *Modeling* or *Monitoring*. The requested data records appear, divided according to request, under the source system they belong to for an InfoSource in the PSA tree.

Features

The data records in BI are transferred to the transfer structure when you load data with the transfer method PSA. One TRFC is performed for each data package. Data is written to the PSA table from the transfer structure, and stored there. A transparent PSA table is created for each transfer structure that is activated. The PSA tables each have the same structure as their respective transfer structures. They are also flagged with key fields for the request ID, the data package number, and the data record number.

Since the requested data is stored unchanged in the PSA, it may contain errors if it contained errors in the source system. If the requested data records have been written to the PSA table, you can check the data for the request and change incorrect data records [Page 51].

Depending on the type of update, data is transferred from the PSA table into the communication structure using the transfer rules. From the communication structure, the data is updated to the corresponding data target.

Using partitioning, you can separate the dataset of a PSA table into several smaller, physically independent, and redundancy-free units. This separation can mean improved performance when you update data from the PSA. In the BW Customizing Implementation Guide, under *Business Information Warehouse → Connections to Other Systems → Maintain Control Parameters for Data Transfer*, you determine the number of data records from which you want to create a partition. Only data records from a complete request are stored in a partition. The specified value is a threshold value.

As of SAP BW 3.0, you can use the PSA to load hierarchies from the DataSources released for this purpose. The corresponding DataSources will be delivered with Plug-In (-A) 2001.2, at the earliest. You can also use a PSA to load hierarchies from files.

Constraints

The number of fields is limited to a maximum of 255 when using TRFCs to transfer data. The length of the data record is limited to 1962 bytes when you use TRFCs.

Data transfer with IDocs cannot be used in connection with the PSA.

4.1.6.4.3.1.1 Types of Data Update with PSA

Prerequisites

You have defined the *PSA* transfer method in the transfer rules maintenance.

Features

Processing options for the PSA transfer method

In contrast to a data request with IDocs, a data request in the PSA also gives you various options for updating data in the BI system. Upon selection, you need to weigh data security against performance for the loading process.

If you create an InfoPackage in the scheduler for BI, you specify the type of data update on the *Processing* tab page.

The following processing options are available in the *PSA* transfer method:

Processing Option	Description	More Information
PSA and Data Targets/InfoObjects in Parallel (By Package)	A process is started to write the data from this data package into the PSA for each data package. If the data is successfully updated in the PSA, a second parallel process is started. In this process, the transfer rules are used for the package data records, data is adopted by the communication structure, and it is finally written to the data targets. Posting of the data occurs in **parallel by package**. This method is used to update data into the PSA and the data targets with a high level of performance. BI receives the data from the source system, writes it to the PSA, and starts the update immediately, in parallel, in the corresponding data target.	The maximum number of processes, which is set in the source system in Maintaining Control Parameters for Data Transfer [Page 536], does not restrict the number of processes in BI. Therefore, many dialog processes in the BI system could be necessary for the loading process. Make sure that enough dialog processes are available in the BI system. If the data package contains incorrect data records, you have several options allowing you to continue working with the records in the request. You can specify how the system should react to incorrect data records. More information: Handling Data Records with Errors [Page 373]. You also have the option of correcting data in the PSA and updating it from here (refer to Checking and Changing Data [Page 51]). Note the following when using transfer and update routines: If you choose this processing option and then request processing takes place in parallel during loading, the global data is deleted because a new process is used for every data package in further processing.

PSA and then to Data Target/InfoObject (by Package)	A process that writes the package to the PSA table is started for each data package. When the data has been successfully updated to the PSA, the same process writes the data to the data targets. The data is posted in **serial by package**. Compared with the first processing option, you have better control over the whole data flow with a serial update of data in packages, because the BI system carries it out using only one process for each data package. Only a certain number of processes are necessary for each data request in the BI system. This number is defined in the settings made in the maintenance of the control parameters in customizing for extractors.	If the data package contains incorrect data records, you have several options allowing you to continue working with the records in the request. More information: Handling Data Records with Errors [Page 373]. You also have the option of correcting data in the PSA and updating it from here (refer to Checking and Changing Data [Page 51]). Note the following when using transfer and update routines: If you choose this processing option and then request processing takes place in parallel during loading, the global data is deleted because a new process is used for every data package in further processing.
Only PSA	Using this method, data is written to the PSA and is not updated any further. You have the advantage of having data stored safely in BI and having the PSA, which is ideal as a persistent incoming data store for mass data as well. The setting for the maximum number of processes in the source system can also have a positive impact on the number of processes in BI. To further update the data automatically in the corresponding data target, wait until all the data packages have arrived and have been successfully updated in the PSA, and select *Update in DataTarget* from the *Processing* tab page when you schedule the InfoPackage in the Scheduler. A process that writes the package to the PSA table is	When using the InfoPackage in a process chain, this setting is hidden in the scheduler. This is because the setting is represented by its own process type in process chain maintenance and is maintained there. Handling Duplicate Data Records [Page 371] (only possible with the processing type *Only PSA*): The system indicates when master data or text DataSources transfer potential duplicate data records for a key into the BI system. The *Ignore Duplicate Data Records* indicator is also set by default in this case. If multiple data records are transferred, the last data record of a request for a particular key is updated in BI by default. Any other data records in the request with the same key are ignored. If

	started for each data package. If you then trigger further processing and the data is updated to the data targets, a process is started for the request that writes the data packages to the data targets one after the other. Posting of the data occurs in **serial by request**.	the *Ignore Duplicate Data Records* indicator is not set, duplicate data records will cause an error. The error message is displayed in the monitor. Note the following when using transfer and update routines: If you choose this processing option and request processing takes place serially during loading, the global data is kept as long as the process with which the data is processed is in existence.

Further updating from the PSA

Several options are available to update the data from the PSA into the data targets.

- To immediately update the request data in the background, select the request in the PSA tree and choose *Context Menu (Right Mouse Button)* → *Start Update Immediately*.
- To schedule a request update using the Scheduler, select the request in the PSA tree and choose *Context Menu (Right Mouse Button)* → *Schedule Update*.

  The *Scheduler (PSA Subsequent Update)* appears. Here you can define the scheduling options for background processing. For data with flexible update, you can also specify and select update parameters where data needs to be updated.
- To further update the data automatically in the corresponding data target, wait until all the data packages have arrived and have been successfully updated in the PSA, and select *Update in DataTarget* from the *Processing* tab page when you schedule the InfoPackage in the Scheduler.

  When using the InfoPackage in a process chain [Page 375], this setting is hidden in the scheduler. This is because the setting is represented by its own process type in process chain maintenance and is maintained there.

Simulating/canceling update from PSA

To simulate the data update for a request using the Monitor, select the request in the PSA tree, and choose *Context menu (right mouse button)* → *Simulate/Cancel update.*

The monitor detail screen appears. On the *Detail* tab page, select one or more data packages and choose *Simulate Update*. In the following screen, define the simulation selections and select *Execute Simulation*. Enter the data records for which you want to simulate the update and choose *Continue*. You see the data in the communication structure format. In the case of data with flexible updating, you can change to the view for data target data records. In the data target screen you can display the records belonging to the communication structure for

selected records in a second window. If you have activated debugging, the *ABAP Debugger* appears and you can execute the error analysis there.

More information: Update Simulation in the Extraction Monitor [Page 797]

Processing several PSA requests at once

To process several PSA requests at once, select the PSA in the PSA tree and choose *Context Menu (Right Mouse Button)* → *Process Several Requests*. You have the option of starting the update for the selected requests immediately or using the scheduler to schedule them. The individual requests are scheduled one after the other in the scheduler. You can delete the selected requests collectively using this function. You can also call detailed information, the monitor, or the content display for the corresponding data target.

During processing, a background process is started for every request. Make sure that there are enough background processes available.

More Information:

Tab Page: Processing [Page 350]

4.1.6.4.3.1.2 Checking and Changing Data

Use

The PSA offers you the option of checking and changing data before you update it further from the PSA table in the communication structure and in the current data target.

You can check and change data records to

- Remove update errors.

  If lower case letters or characters that are not permitted have been used in fields, you can remove this error in the PSA.

- Validate data.

  For example, if, when matching data, it was discovered that a customer should have been given free delivery for particular products, but the delivery had in fact been billed, then you can change the data record accordingly in the PSA.

Prerequisites

You have determined the PSA transfer method in transfer rule maintenance for an InfoSource, and have loaded data into the PSA.

Procedure

You have two options for checking and changing the data:

1. You can edit the data directly.

a. In the PSA tree in the Administrator Workbench, select the request for which you want to check the data and choose *Context menu (secondary mouse button)* → *Edit Data*.

You get to a dialog box where you can choose which data packet and which data records for this packet you want to edit.

b. When you have made your selections choose *Continue*.

You get to the request data maintenance screen.

c. Select the records you want to edit, select *Change,* and enter the correct data. Save the edited data records.

2. Since the data is stored in a transparent database table in the dictionary, you can change the data using ABAP programming with PSA-APIs [Page 52]. Use the programming for a PSA-API with complex data checks or changes to the data that occur regularly.

If you change the number of records for a request in the PSA, thereby adding or deleting records, a correct record count in the BI monitor is no longer guaranteed when posting or processing a request.

Therefore, we recommend not changing the number of records for a request in the PSA.

Result

The corrected data is now available for continued updates.

4.1.6.4.3.1.3 Checking and Changing Data Using PSA-APIs

Use

To perform complex checks on data records, or to carry out specific changes to data records regularly, you can use delivered function modules (PSA-APIs) to program against a PSA table.

If you want to execute data validation with program support, select *Tools* → *ABAP Workbench* → *Development* → *ABAP Editor* and create a program [External].

If you use transfer routines [Page 281] or update routines [Page 317] it may be necessary to read data in the PSA table afterwards.

Employee bonuses are loaded into an InfoCube and sales figures for employees are loaded into a PSA table. If an employee's bonus is to be calculated in a routine in the transformation, in accordance with his/her sales, the sales must be read from the PSA table.

Procedure

1. Call up the function module **RSSM_API_REQUEST_GET** to get a list of requests with request ID for a particular InfoSource of a particular type. You have the option of restricting request output using a time restriction and/or the transfer method.

You must know the request ID, as the request ID is the key that makes managing data records in the PSA possible.

2. With the request information received so far, and with the help of the function module, you can
   - read RSAR_ODS_API_GET data records from the PSA table
   - write RSAR_ODS_API_PUT changed data records in the PSA table.

RSAR_ODS_API_GET

You can call up the function module RSAR_ODS_API_GET with the list of request IDs given by the function module RSSM_API_REQUEST_GET. The function module RSAR_ODS_API_GET no longer recognizes InfoSources on the interface, rather it recognizes the request IDs instead. With the parameter I_T_SELECTIONS, you can restrict reading data records in the PSA table with reference to the fields of the transfer structure. In your program, the selections are filled and transferred to the parameter I_T_SELECTIONS.

The import parameter causes the function module to output the data records in the parameter E_T_DATA. Data output is unstructured, since the function module RSAR_ODS_API_GET works generically, and therefore does not recognize the specific structure of the PSA. You can find information on the field in the PSA table using the parameter E_T_RSFIELDTXT.

RSAR_ODS_API_PUT

After merging or checking and subsequently changing the data, you can write the altered data records into the PSA table with the function module RSAR_ODS_API_PUT. To be able to write request data into the table with the help of this function module, you have to enter the corresponding request ID. The parameter E_T_DATA contains the changed data records.

Result

The corrected data is now available for continued updates.

4.1.6.4.3.1.4 Versioning

Use

If you make an incompatible change to the transfer structure (for example, length changes or the deletion of fields), a version is assigned to the PSA table.

Features

When the system detects an incompatible change to the transfer structure, a new version of the PSA, meaning a new PSA table, is created. Data is written to the new table when the next request is updated.

The original table remains unchanged and is given a version. You can continue to use all of the PSA functions for each request that was written to the old table.

Data is read from a PSA table in the appropriate format.

- If the request was written to the PSA table before the transfer structure was changed, the system uses the format that the transfer structure had before the change.
- If the request was been written to the PSA table after the transfer structure was changed, the system uses the format that the transfer structure has after the change.

If you program against function module RSAR_ODS_API_GET, you can determine that the data is read into the structure of the current version from an old version using parameter I_CURRENT_DATAFORMAT.

4.1.6.4.3.1.5 DB Storage Parameters

Use

You can maintain database storage parameters for PSA tables, master data tables, InfoCube fact- and dimension tables, and DataStore object tables.

Use this setting to determine how the system handles the table when it creates it in the database:

- Use *Data Type* to set in which physical database area (tablespace) the system is to create the table. Each data type (master data, transaction data, organization- and Customizing data, customer data) has its own physical database area, in which all tables assigned to this data type are stored. If selected correctly, your table is automatically assigned to the correct area when it is created in the database.

  We recommend you use separate tablespaces for very large tables.

  You can find information about creating a new data type in SAP Note 0046272 (Introduce new data type in technical settings).

- Via *Size Category*, you can set the amount of space the table is thought to need in the database. Five categories are available in the input help. You can also see here how many data records correspond to each individual category. When creating the table, the system reserves an initial storage space in the database. If the table later requires more storage space, it obtains it as set out in the size category. Correctly setting the size category prevents there being too many small extents (save areas) for a table. It also prevents the wastage of storage space when creating extents that are too large.

You can use the maintenance for storage parameters to better manage databases that support this concept.

You can find additional information about the data type and size category parameters in the ABAP Dictionary table documentation, under Technical Settings [External].

PSA Table

For PSA tables, you access the database storage parameter maintenance by choosing *Goto* → *Technical Attributes* in DataSource maintenance. In dataflow 3.x, you access this setting in transfer rule maintenance in the *Extras* menu.

You can also assign storage parameters for a PSA table already in the system. However, this has no effect on the existing table. If the system generates a new PSA version (a new PSA table) due to changes to the DataSource, this is created in the data area for the current storage parameters.

InfoObject Tables

For InfoObject tables, the maintenance of database storage parameters can be found in the InfoObject maintenance, in the *Extras* menu.

InfoCube/aggregate fact and dimension tables

For fact- and dimension tables, the maintenance of database storage parameters can be found in the InfoCube maintenance, in the *Extras* menu.

DataStore Object Tables (Activation Queue and Table for Active Data)

For DataStore object tables, the maintenance of database storage parameters can be found in DataStore object maintenance, in the *Extras* menu.

4.1.6.4.3.1.6 Reading the PSA and Updating a Data Target

Use

You can use this process to further update data from the PSA. This takes place after all data packages arrived in the PSA and were successfully updated there.

Note that it is not possible to create more than one process of type *Read PSA and Update Data Target* for one request or InfoPackage at any one time. You cannot simultaneously update into more than one data target. Updating into more than one data target can currently only occur sequentially.

This process replaces the indicator *Subsequently Update into Data Targets* on the *Processing* tab page in the Scheduler. When using an InfoPackage in a process chain, this indicator is grayed out in the Scheduler and the *Read PSA and Update Data Target* process is controlled by process chain maintenance.

Any settings previously made in the InfoPackage are then ignored.

Procedure

1. In the SAP BW Menu, choose *Administration* → *Process Chains*.

   In the *Administrator Workbench,* choose *Process Chain Maintenance* from the symbol bar.

   The *Process Chain Maintenance Planning View* screen appears.

2. In the left-hand screen area of the required Display Component, navigate to the process chain in which you want to insert the process. Double-click to select it. Alternatively, you can create a new process chain.

   The system displays the process chain plan view in the right-hand side of the screen. You can find additional information under Creating a Process Chain [Page 402].

3. In the left-hand screen area, choose *Process Types.*

   The system now displays the process categories available.

4. Insert the *Read PSA and Update Data Target* application process into the process chain using Drag&Drop. The dialog box for inserting a process variant appears.

5. In the *Process Variants* field, enter the name of the application process you want to insert into the process chain. A value help is available, which lists all process variants that have already been created.

   Choose *Create* if you want to create a new process variant. A dialog box appears, in which you can enter a description for your application process.

   Enter the description for your application process and choose *Next.* The process chain maintenance screen appears.

   In the upper screen area, the system displays the following information for the variant:

   - Technical name
   - Description (You can make an entry in this field)
   - Last Changed by
   - Last changed on

6. There are two ways of specifying which requests are to be further updated into which data targets:

   a. In the table, in the *Object Type* column, you can choose *Execute InfoPackage,* and then one or more InfoPackages to be included in the process chain. Select **neither** *PSA Table* **nor** *Data Target.*

   As a result, during the chain run, those requests are updated that were loaded with the specified InfoPackages into the PSA within the chain. Data targets and PSA tables are stored in the InfoPackages.

   b. Select *PSA Table* and *Data Target.* You can also choose *Request* as the Object Type in the table, and then one or more requests.

      As a result, only the selected requests are updated from the specified PSA table into the specified data target.

      Only use this setting when calling up the process for the first time. Afterwards, the request is already in the data target and must then be deleted before updating again. Furthermore, this setting cannot be transported as the requests numbers are local to the system and the specified request definitely does not exist in the target system.

7. Save your entries and go back. The *Process Chain Maintenance Planning View* screen appears.

Result

You have inserted the *Read PSA and Update Data Target* application process into the process chain.

You can find further information about the additional steps taken when creating a process chain here: Creating a Process Chain [Page 402].

4.1.6.5 nfoObject

Definition

Business evaluation objects are known in BI as InfoObjects. They are divide into characteristics (for example, customers), key figures (for example, revenue), units (for example, currency, amount unit), time characteristics (for example, fiscal year) and technical characteristics (for example, request number).

Use

InfoObjects are the smallest units of BI. Using InfoObjects, information is mapped in a structured form. This is required for constructing InfoProviders.

InfoObjects with attributes or texts can themselves also be InfoProviders [Page 125] (if in a query).

Structure

Characteristics are sorting keys, such as company code, product, customer group, fiscal year, period, or region. They specify classification options for the dataset and are therefore reference objects for the key figures. In the InfoCube, for example, characteristics are stored in dimensions. These dimensions are linked by dimension IDs to the key figures in the fact table. The characteristics determine the granularity (the degree of detail) at which the key figures are kept in the InfoCube. In general, an InfoProvider contains only a sub-quantity of the characteristic values from the master data table. The master data includes the permitted values for a characteristic. These are known as the characteristic values.

The **key figures** provide the values that are reported on in a query. Key figures can be quantity, amount, or number of items. They form the data part of an InfoProvider.

Units are also required so that the values for the key figures have meanings. Key figures of type amount are always assigned a currency key and key figures of type quantity also receive a unit of measurement.

Time characteristics are characteristics such as date, fiscal year, and so on.

Technical characteristics have only one organizational meaning within BI. An example of this is the request number in the InfoCube, which is obtained as ID when loading requests. It helps you to find the request again.

Special features of characteristics:

If characteristics have attributes, texts, or hierarchies at their disposal then they are referred to as master data-bearing characteristics [Page 113]. Master data is data that remains unchanged over a long period of time. Master data contains information that is always needed in the same way. References to this master data can be made in all InfoProviders. You also have the option of creating characteristics with references. A reference characteristics provides the attributes, master data, texts, hierarchies [External], data type, length, number and type of compounded characteristics, lower case letters and conversion routines for new characteristics.

A hierarchy is always created for a characteristic. This characteristic is the basic characteristic for the hierarchy (basic characteristics are characteristics that do not reference other characteristics). Like attributes, hierarchies provide a structure for the values of a characteristic. *Company location* is an example of an attribute for *Customer*. You use this, for example, to form customer groups for a specific region. You can also define a hierarchy to make the structure of the *Customer* characteristic clearer.

Special features of key figures:

A key figure is assigned additional properties that influence the way that data is loaded and how the query is displayed. This includes the assignment of a currency or unit of measure, setting aggregation and exception aggregation, and specifying the number of decimal places in the query.

Integration

InfoObjects can be part of the following objects:

1. **Component of an InfoSource**

   An InfoSource is a quantity of InfoObjects that logically belong together and are updated in InfoProviders.

2. **Composition of an InfoProvider:**

   An InfoProvider consists of a number of InfoObjects.

   In an InfoCube, the characteristics, units, and time characteristics form the basis of the key fields, and the key figures form the data part of the fact table of the InfoCube.

   In a DataStore object, characteristics generally form the key fields, but they can also be included in the data part, together with the key figures, units and time characteristics.

3. **Attributes for InfoObjects**

4.1.6.5.1 nfoObject Catalog

Definition

An InfoObject catalog is a collection of InfoObjects grouped according to application-specific criteria. There are two types of InfoObject catalogs: ***Characteristic*** and ***Key figure.***

Use

An InfoObject catalog is assigned to an InfoArea.

An InfoObject catalog is an organizational aid and is not for intended for data analysis purposes.

For example, all the InfoObjects that are used for data analysis in the area of *Sales and Distribution* can be grouped together in one InfoObject catalog. This makes it much easier for you to handle what might turn out to be a very large number of InfoObjects for any given context.

An InfoObject can be included in several InfoObject catalogs.

In InfoProvider definition, you can select an InfoObject catalog as a filter for the template.

4.1.6.5.1.1 Creating InfoObject Catalogs

Prerequisites

Ensure that all the InfoObjects that you want to transfer into the InfoObject catalog are active. If you want to define an InfoObject catalog in the same way as an InfoSource, then the InfoSource has to be available and active.

Procedure

1. Create an InfoArea, to which you want to assign the new InfoObject catalog. This function is on the first level of the hierarchy in the Administrator Workbench, under InfoObjects.
2. Use the right mouse button to create an InfoObject catalog in the InfoArea. If you want to make a copy of an existing InfoObject catalog, specify a reference InfoObject catalog.
3. Choose either *characteristic* or *key figure* for the InfoObject type, and choose *Create*.
4. Transferring InfoObjects:

   On the left side of the screen there are various templates to choose from. These allow you to get a better overview in relation to a particular task. For performance reasons, the default setting is an empty template. Using the pushbuttons, select an InfoSource (only the InfoObjects for the communication structure of the InfoSource are displayed), an InfoCube, a DataStore object, an InfoObject catalog or all InfoObjects.

   On the right side of the screen you compile your InfoObject catalog. Transfer the desired InfoObjects into the InfoObject catalog using Drag&Drop You can also simultaneously select multiple InfoObjects.
5. Activate the InfoObject catalog.

4.1.6.5.1.2 Additional InfoObject Catalog Functions

Documents

This function allows you to display, create or change documents for your InfoObject catalog.

See: Documents [External].

Info Functions

There are various info-functions on the status of the InfoCube Catalog:

- the log display for activation and deletion runs of the InfoObject Catalog,
- the current system settings, the object catalog entry.

Display in Tree

You can use this function to display all properties of your InfoObject catalog in a concise hierarchical structure.

Version Comparison

You use this function to compare the following InfoObject catalog versions:

- the active and revised versions of an InfoObject catalog
- the active and Content versions of an InfoObject catalog

- the revised and Content versions of an InfoObject catalog

In this way you are able to compare all properties.

Transport Connection

You can transport the InfoObject catalog. All BW Objects that are needed to ensure a consistent status in the target system are collected automatically.

Where-Used List

You can determine which other objects in BW use this InfoObject catalog. You can determine what effects making a particular change in a particular way will have, and whether this change is permitted at the moment or not.

InfoObject Maintenance

You get to the transaction for displaying, creating, and changing InfoObjects from *Extras* in the main menu.

4.1.6.5.2 InfoObject Naming Conventions

Use

As is the case for other objects in BI, the customer namespace A-Z is also reserved for InfoObjects.

When you create an InfoObject, the name you give it has to begin with a letter. BI Content InfoObjects start with 0.

For more information about namespaces, see Namespaces for BI Objects [Page 18].

Integration

If you change an InfoObject in the SAP namespace, your modified InfoObject is not overwritten immediately when you install a new release, and your changes remain in place.

BI Content InfoObjects are initially delivered in the D version. If you use the BI Content InfoObject, it is activated. If you change the activated InfoObject, a new M version is generated. When this M version is activated, it overwrites the previous active version.

When you are determining naming conventions for InfoObjects, keep in mind that the length of an InfoObject is restricted to 60 characters. If additional characteristics are compounded to other InfoObjects, the length is the concatenated value. See also Tab Page: Compounding [Page 85].

4.1.6.5.3 Creating InfoObjects: Characteristics

Procedure

1. In the context menu of your InfoObject catalog for characteristics, select *Create InfoObject*.

2. Enter a name and a description
3. Specify a reference characteristic or a template InfoObject. If you choose a template InfoObject, you copy its properties and use them for the new characteristic. You can edit the properties as required. For more information about reference characteristics, see Tab Page: Compounding [Page 85] in the *Ref*erence InfoObject section.
4. Confirm your entries.
5. Maintain Tab Page: General [Page 61]. You have to enter a description, data type and data length. The following settings and tab pages are optional.

Maintain Tab Page: Business Explorer [Page 63]

Maintain Tab Page: Master Data/Texts [Page 80]

Maintain Tab Page: Hierarchy [Page 83]

6. Maintain Tab Page: Attributes [Page 83]. This tab page is only available if you have set the *With Master Data* indicator on the *Master Data/Texts* tab page.

Maintain Tab Page: Compounding [Page 85]

7. *Save* and *Activate* the characteristic you have created.

Before you can use characteristics, they have to be *activated*.

If you choose *Save*, the system creates all the characteristics that have been changed and saves the table entries. However, they cannot be used for reporting in InfoProviders yet. If there is an older *active* version, this is retained initially.

The system only creates the relevant objects created in the data dictionary (data elements, domains, text tables, master data tables, and programs) after you have *activated* the characteristics. Only then do the InfoProviders use the activated, new version.

In InfoObject maintenance, you can switch between any D, M, or A versions that exist for an InfoObject at any time.

4.1.6.5.3.1 ab Page: General

Use

On this tab page, you specify the basic properties of the characteristic.

Structure

Dictionary

Specify the *data type* and the *data length*. The system provides you with selection options using F4 Help.

The following data types are supported for characteristics:

Char:	Numbers and letters	Character length 1 - 60
Numc:	Numbers only	Character length 1 - 60

Dats:	Date	Character length 8
Tims:	Time	Character length 6

Miscellaneous

Lowercase Letters Allowed / Not Allowed

If this indicator is set, the system differentiates between lowercase letters and capital letters when you use a screen template to input values. If this indicator is not set, the system converts all the letters into capital letters when you use a screen template to input values. During the load process or in the transformation, no conversion occurs. This means that values with lowercase letters cannot be updated to an InfoObject that does not allow lowercase letters.

If you choose to allow the use of lowercase letters, you must be aware of what happens when you input variables:

If you want to use the characteristic in variables [External], the system is only able to find the values for the characteristic if the lowercase letters and the capital letters are typed in accurately on the input screen for variables. If, on the other hand, you do not allow the use of lowercase letters, any characters that you type in the variable screen, are converted automatically into capital letters.

Conversion Routines

The standard conversion for the characteristic is displayed. If this standard conversion is unsuitable, you can override it by specifying a conversion routine in the underlying domain. See also Conversion Routines in BI Systems [Page 98].

Attribute Only

If you select *Attribute Only*, the created characteristic can be used only as a display attribute for another characteristic and not as navigation attribute. Furthermore, you cannot transfer the characteristic into InfoCubes. However, you can use it in DataStore objects or InfoSets.

Characteristic Is Document Property

You can specify that a characteristic be used as a document property. This enables you to assign a comment (this can be any document) to a combination of characteristic values. See also Documents [External] and the example Characteristic Is Document Property [External].

Since it does not make sense to use this comment function for all characteristics, you need to identify explicitly the characteristics that you want to appear in the comments.

If you set this indicator, the system generates a property (attribute) for this characteristic in the metamodel of the document management system. For technical reasons, this property (attribute) has to be written to a (dummy) transport request (the appropriate dialog box appears) but it is not actually transported.

Constants

By assigning a *constant* to a characteristic, you give it a fixed value. The characteristic then exists on the database side (for example, verifications), but it does not appear in reporting. Assigning a constant is most useful with compounded characteristics.

The storage location characteristic is compounded with the plant characteristic. If only one plant is ever run within the application, a constant can be assigned to the plant. The verification for the storage-location master table runs correctly with this value for the plant. In the query, however, only the storage location appears as a characteristic.

Special Case:

If you want to assign the SPACE constant (type CHAR) or 00..0 (type NUMC) to the characteristic, type # in the first position.

Transfer Routine

When you create a transfer routine, it is valid globally for the characteristic, and is included in all the transformation rules that contain the InfoObject.

During the data transfer, first the logic stored in the individual transformation rule is executed. Then the transfer routine for the value for the corresponding field is executed for each InfoObject that has a transfer routine.

The transfer routine can thus store InfoObject-dependent coding that only needs to be maintained once, but that is valid for all the transformation rules.

4.1.6.5.3.2 Tab Page: Business Explorer

Use

On this tab page you determine the properties that are required in the Business Explorer for reporting on or analyzing characteristics.

Structure

General Settings

You can make the following settings for the InfoObjects contained in the InfoProvider on an InfoProvider by InfoProvider basis. The settings are only valid in the relevant InfoProvider. See also Additional Functions in InfoCube Maintenance [Page 133] and Additional Functions in DataStore Object Maintenance [Page 158].

Layout

For characteristics with texts: Under *Display*, you select whether you want to display text in the Business Explorer and if yes, which text. You can choose from the following display options: *No Display, Key, Text, Key and Text,* or *Text and Key*. This setting can be overwritten in queries.

Text Type

For characteristics and texts: In this field you set whether you want to display short, medium or long text in the Business Explorer.

Description BEx

In this field, you determine the description that appears for this characteristic in the Business Explorer. You choose between the long and short descriptions of the characteristic. This

setting can be overwritten in queries. For more information, see Priority Rule with Formatting Settings [External].

Selection

The selection describes if and how the characteristic values have to be restricted in queries. If you choose the *Unique for Every Cell* option, the characteristic must be restricted to one value in each column and in each structure of all the queries. You cannot use this characteristic in aggregates. Typical examples of this kind of characteristic are *Plan/Actual ID* or *Value Type*.

Filter Selection in Query Definition

This field describes how the selection of filter values or the restriction of characteristics is determined when you define a query.

When you restrict characteristics, the values from the master data table are usually displayed. For characteristics that do not have master data tables, the values from the SID Table [External] are displayed instead. In many cases it is more useful to only display those values that are also contained in an InfoProvider. Therefore you can also choose the setting *Only Values in InfoProvider*.

Filter Selection in Query Execution

This field tells you how the selection of filter values is determined when a query is executed.

When queries are executed, the selection of filter values is usually determined by the data that is selected by the query. This means that only the values for which data has been selected in the current navigation status are displayed.

In many cases, however, it can be useful to include additional values. Therefore you can also choose the settings *Only Values in InfoProvider* and *Values in Master Data Table*. If you make this selection, however, you may get the message “No data found” when you select your filter values.

These settings for input help can also be overwritten in the query. For more information, see Priority Rule with Formatting Settings [External].

Base Unit of Measure

You specify a unit InfoObject of type unit of measure. The unit InfoObject must be an attribute of the characteristic. This unit InfoObject is used when quantities are converted for the master data-bearing characteristic in the Business Explorer.

Currency Attribute

You select a unit InfoObject of type currency that you have created as an attribute for the characteristic. In this way, you can define variable target currencies in the currency translation types. The system determines the target currency using the master data when you perform currency translation in the Business Explorer [External] or load dynamically. See the example Determining the Target Currency from an InfoObject [External].

Authorization Relevance

You choose whether a particular characteristic is included in the authorization check when you are working with the query.

Mark a characteristic as *authorization-relevant* if you want to create authorizations that restrict the selection conditions for this characteristic to single characteristic values.

You can only mark the characteristic as *Not Authorization-Relevant* if it is no longer being used as a field for the authorization object.

See also:

Analysis Authorizations [External]

BEx Map

Geographical Type

For each geo-relevant characteristic [Page 67] you have to specify a geographical type. There are four options to choose from.

1. *Static geo-characteristic*: For this type you can use shape files (country borders, for example), to display the characteristic on a map in the Business Explorer.
2. *Dynamic Geo-Characteristic*: For this type geo-attributes are generated that make it possible, for example, to display customers as a point on a map.
3. *Dynamic Geo-Characteristic with Attribute Values*: For this type the geo-attributes of a geo-characteristic of type 2, which is an attribute, are used.
4. *Static geo-characteristic with geo-attributes*: Just like static geo-characteristics, with the addition of generated geo-attributes.

See also Static and Dynamic Geo-Characteristics [Page 67].

If you choose the *Not a Geo-Characteristic* option, this characteristic cannot be used as a geo-characteristic for displaying information on the BEx Map. Geographical attributes of the InfoObject (such as 0LONGITUDE, 0ALTITUDE) are deleted.

Geographical Attribute

If you have selected the *Dynamic Geo-Characteristic with Attribute Values* geographical type for the characteristic, on this tab page you specify the characteristic attribute whose geo-attributes you want to use.

Uploading Shapefiles

For static geo-characteristics: Use this function to upload the geo-information files that are assigned to the characteristic. These files are stored in the BDS as files that logically belong to the characteristic.

See also Shapefiles [Page 68].

Downloading Geo-Data

For dynamic geo-characteristics: You use this function to load the master data for a characteristic to your PC, where you can use your GIS tool to geocode the data. You use a flat file to load the data again as a normal data load into the relevant BI master data table.

4.1.6.5.3.2.1 Mapping Geo-Relevant Characteristics

Definition

To display BI data geographically, a link between this data and the respective geographical characteristic must be created. This process is called *Mapping Geo-Relevant Characteristics.*

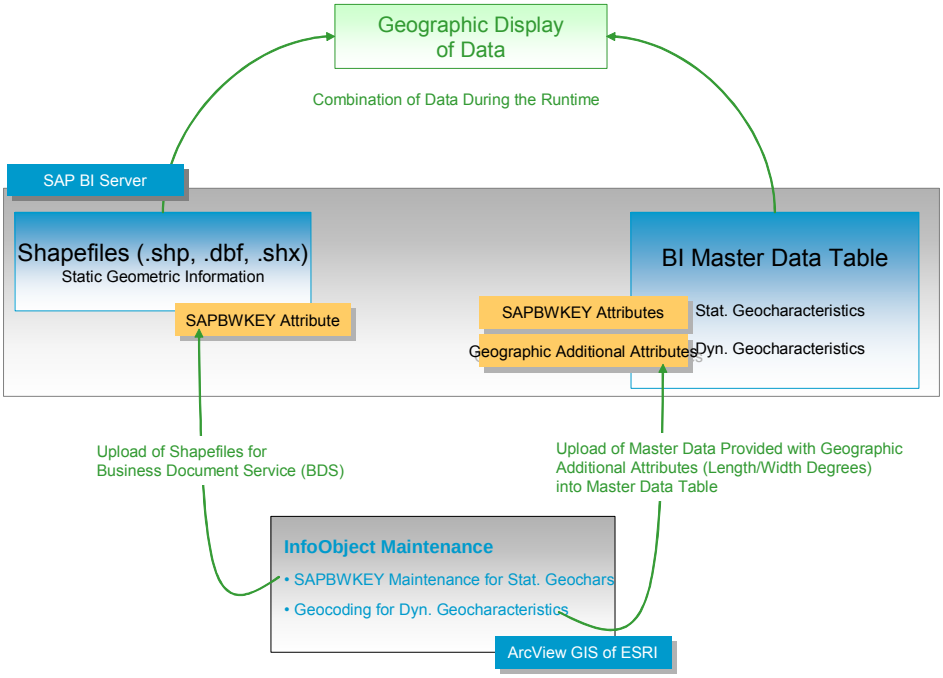
Structure

The geographical information about geographical boundaries of areas that are displayed using static geo-characteristics is stored in Shapefiles [Page 68]. In the Shapefile, a BI-specific attribute called the SAPBWKEY is responsible for connecting an area on the map with the corresponding characteristic value in BI. This attribute matches the characteristic value in the corresponding BI master data table. This process is called **SAPBWKEY Maintenance for Static Geo-Characteristics** . See SAPBWKEY Maintenance for Static Geo-Characteristics [Page 71]

You can use *ArcView GIS* or other software that has functions for editing dBase files to carry out the SAPBWKEY maintenance (MS Excel, for example).

With data in point form that is displayed using dynamic geo-characteristics, geographical data is added to BI master data. The process of assigning geographical data to entries in the master data table is called **geocoding**. See Geocoding [Page 76]

The software *ArcView GIS* from ESRI (Environmental Systems Research Institute) geocodes the InfoObjects.



Integration

You can execute the geocoding with the help of the *ArcView GIS from ESRI* software. As well as geocoding, ArcView also offers a large number of functions for special, geographical problems that are not covered by SAP NetWeaver Business Intelligence. ArcView enables you to create your own maps, for example, a map of your sales regions. For more detailed information, see the ArcView documentation.

When you buy *SAP NetWeaver BI,* you receive a voucher that you can use to order *ArcView GIS* from ESRI.

The installation package also contains a CD developed specially by SAP and ESRI. The CD contains a range of maps covering the whole world in various levels of detail. All maps on this data CD are optimized already for use with SAP NetWeaver BI. The .dbf files for the maps already contain the column SAPBWKEY that is predefined with default values. For example, the world map (cntry200) contains the usual values from the SAP system for countries in the SAPBWKEY column. Therefore, you can use the map immediately to evaluate your data geographically. You do not have to maintain the SAPBWKEY.

You can get additional detailed maps in ESRI Shapefile format from ESRI.

4.1.6.5.3.2.1.1 Static and Dynamic Geo-Characteristics

Definition

Static and dynamic characteristics describe data with a geographical reference (for example, characteristics such as customer, sales region, country). Maps are used to display and evaluate this geo-relevant data.

Structure

There are four different types of geo-characteristic:

1. **Static geo-characteristics**

   A static geo-characteristic is a characteristic that describes a surface (polygon), whose geographical coordinates rarely change. Country and region are examples of static geo-characteristics.

   Data from areas or polygons are stored in Shapefiles [Page 68] that define the geometry and the attributes of the geo-characteristics.

2. **Dynamic geo-characteristics**

   A dynamic geo-characteristic is a characteristic that describes a location (information in point form), whose geographical coordinates can change more frequently. Customer and plant are examples of dynamic geo-characteristics because they are rooted to one geographical “point” that can be described by an address, and the address data of these characteristics can often change.

   A range of standard attributes are added to this geo-characteristic in SAP NetWeaver BI. These standard attributes store the geographical coordinates of the corresponding object for each row in the master data table. The geo-attributes concerned are:

Technical Name	Description	Data Type	Length
LONGITUDE	Longitude of the location	DEC	15
LATITUDE	Latitude of the location	DEC	15
ALTITUDE	Altitude of the location (height above sea level)	DEC	17
PRECISID	Identifies how precise the data is	NUMC	4

SRCID	ID for the data source	CHAR	4

At present, only the LONGITUDE and LATITUDE attributes are used. ALTITUDE, PRECISID and SRCID are reserved for future use.

If you reset the geographical type of a characteristic to *Not a Geo-Characteristic*, these attributes are deleted in the InfoObject maintenance.

3. **Dynamic geo-characteristics with values from attributes**

   To save you having to geocode each dynamic geo-characteristic individually, a dynamic geo-characteristic can get its geo-attributes (longitude, latitude, altitude) from another dynamic characteristic that has been geocoded already (postal code, for example). Customers and plants are examples of this type of *dynamic geo-characteristics with values from attributes* (type 3).

   The system treats this geo-characteristic as a regular dynamic geo-characteristic that describes a location (geographical information as a point on map). The geo-attributes described above are not added to the master data table on the database level. Instead, the geo-coordinates are stored in the master data table of a regular attribute of the characteristic.

   You want to define a dynamic geo-characteristic for *Plant* with the postal code as an attribute. The geo-coordinates are generated from the postal code master data table during the runtime.

   This method prevents redundant entries from appearing in the master data table.

4. **Static geo-characteristics with geo-attributes**

   A static geo-characteristic that includes geo-attributes (longitude, latitude, altitude) which geo-characteristics of type 3 are able to refer to. The postal code, for example, can be used as a *static geo-characteristic with geo-attributes*.

   0POSTCD_GIS (postal code) is used as an attribute in the dynamic geo-characteristic 0BPARTNER (business partner) that gets its geo-coordinates from this attribute. In this way, the location information for the business partner is stored on the level of detail of the postal code areas.

See also:

Delivered Geo-Characteristics [Page 70]

4.1.6.5.3.2.1.1.1 Shapefiles

Definition

ArcView GIS software files from ESRI that contain digital map material of areas or polygons (shapes). Shapefiles define the geometry and attributes of static geo-characteristics. Note that shapefiles have to be available in the format of the World Geodetic System 1984 (WGS

84). For more information on the World Geodetic System WGS 84, see **www.wgs84.com.**

Use

Shapefiles serve as a basis for displaying BI data on maps.

Structure

Format

The format of ArcView shapefiles uses the following files with special file enhancements:

.dbf – dBase file that saves the attributes or values of the characteristic

.shp – saves the current geometry of the characteristic

.shx – saves an index for the geometry

These three files are saved for each static geo-characteristic in the Business Document Service (BDS) and loaded to the local computer from BDS when you use BEx Map.

Shapefile Data from the ESRI BI Mapping Data CD

The map data from the ESRI BI mapping data CD was chosen as the basic reference data level to provide you with a detailed map display and thematic mapping material at the levels of world maps, continents and individual countries. The reference data levels involve country boundaries, state boundaries, towns, streets, railways, lakes and rivers. The mapping data is geographically subdivided into data for 21 separate maps.

There is mapping data for:

- a world map
- seven maps on continent level, for example, Asia, Europe, Africa, North America, South America.
- 13 maps on country level: How current the data for the countries is varies. Most of the country boundaries are as they were between 1960-1988, some countries have been updated to their position in 1995.

The names of the shapefiles on the ESRI BI mapping data CD follow a three-part naming convention.

- The first part consists of an abbreviation of the thematic content of the shapefile, for example, *cntry* stands for a shape file with country boundaries.
- The second part of the name indicates the level of detail. There are, for example, three shapefiles with country boundary information at different levels of detail. The least detailed shapefile begins with *cntry1*, whereas the most detailed shapefile begins with *cntry3*.
- The third part of the name indicates the version number of the shapefile, based on the last two digits of the year beginning with the year 2000. Therefore, the full name of the shapefile with the most detailed country boundary information is *cntry300.*

All shapefiles on the ESRI BI mapping data CD already contain the SAPBWKEY column. For countries, the two-figure SAP country key is entered in the SAPBWKEY column.

The *Readme.txt* file on the ESRI BI mapping data CD contains further, detailed information on the delivered shapefiles, the file name conventions used, the

mapping data descriptions and specifications, data sources, and how up-to-date the data is.

Integration

At run time, the shapefiles are downloaded from the BI system to the IGS (Internet Graphic Server). The files are copied into the *../data/shapefiles* directory. If a specific shapefile is already in this directory, it is not copied again. If in the meantime, the shapefile has been changed in the Business Document Service (BDS), the latest version is automatically copied into the local directory.

Depending on the level of detail, shapefiles can be quite large. The shapefile cntry200.shp with the country boundaries for the entire world is around 2.2 megabytes. For smaller organizational units, such as federal states, the geometric information is saved in multiple shapefiles. You can assign a characteristic to several shapefiles (for example, federal states in Germany, France and so on).

4.1.6.5.3.2.1.1.2 Delivered Geo-Characteristics

Definition

With Business Content, SAP NetWeaver BI delivers a range of geo-characteristics.

Structure

The following are the most important delivered geo-characteristics:

Static geo-characteristics

Technical Name	Description
0COUNTRY	Country key
0DATE_ZONE	Time zone
0REGION	Region (federal state, province)

Dynamic geo-characteristics

Technical Name	Description
0APO_LOCNO	Location number
0TV_P_LOCID	IATA location

Dynamic geo-characteristics with values from attributes

Technical Name	Attributes	Description
0BPARTNER	0POSTCD_GIS	Business partner
0CONSUMER	0POSTCD_GIS	Consumer
0CUSTOMER	0POSTCD_GIS	Customer number

0PLANT	0POSTCD_GIS	Plant
0VENDOR	0POSTCD_GIS	Vendor

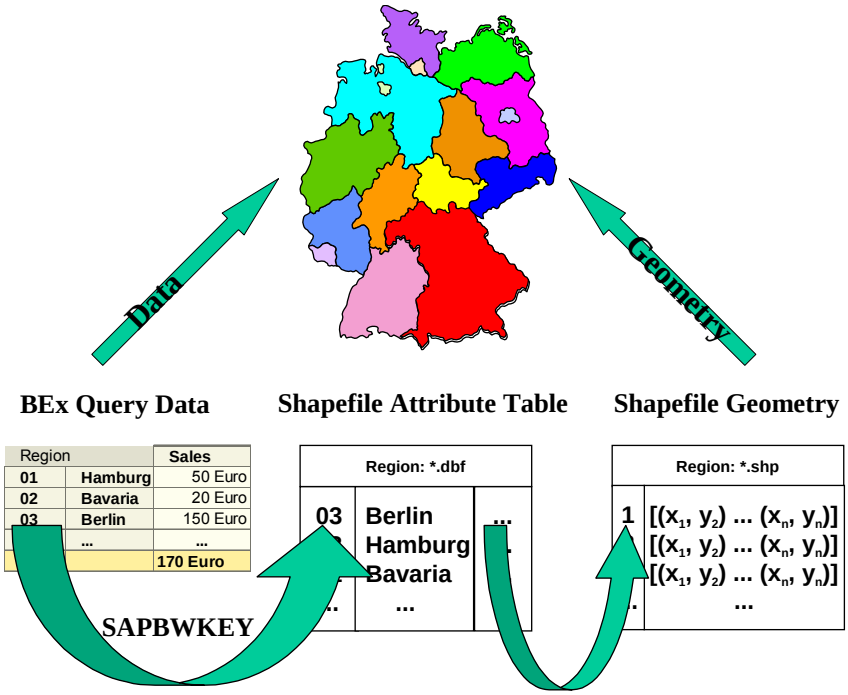
Static geo-characteristics with geo-attributes

Technical Name	Description
0CITYP_CODE	City district code for city and street file
0CITY_CODE	City code for city and street file
0POSTALCODE	Postal/zip code
0POSTCD_GIS	Postal code (geo-relevant)

4.1.6.5.3.2.1.2 SAPBWKEY Maintenance for Static Geo-Characteristics

Purpose

During run time, BI data is combined with a corresponding Shapefile [Page 68]. This enables the BI data to be displayed in geographical form (country, region, and so on) using color shading, bar charts, or pie charts. The SAPBWKEY makes sure that the BI data is assigned to the appropriate Shapefile.



In the standard Shapefiles delivered with the ESRI BI map CD, the SAPBWKEY column is already filled with the two-character SAP country keys (DE, EN, and

so on). You can use these Shapefiles without having to maintain the SAPBWKEY beforehand.

Prerequisites

You have marked the geo-relevant characteristic as geo-relevant in the InfoObject maintenance.

Before you are able to follow the example that explains how you maintain the SAPBWKEY for static geo-characteristics, you must ensure that SAP DemoContent is active in your BI system.

You can use ArcView GIS from ESRI to maintain the SAPBWKEY, or you can use other software (MS Excel or FoxPro, for example) that has functions for displaying and editing dBase files.

Process Flow

For static geo-characteristics (such as *Country* or *Region)* that represent the geographical drilldown data for a country or a region, you have to maintain the SAPBWKEY for the individual country or region in the attributes table of the Shapefile. The attributes table is a database table stored in dBase format. Once you have maintained the SAPBWKEY, you load the Shapefiles (.shp, .dbf, .shx) into BI. The Shapefiles are stored in the Business Document Service (BDS), a component of the BI server.

The following section uses the example of the 0D_COUNTRY characteristic to describe how you maintain the SAPBWKEY for static geo-characteristics. You use the CNTRY200 Shapefile from the ESRI BI map data CD. The CD contains the borders of all the countries in the world. The maintenance of the SAPBWKEY for static geo-characteristics consists of the following steps.

1. You create a local copy of the Shapefile from the BI data CD (.shp,.shx,.dbf) [Page 73].
2. You download BI master data into a dBase file [Page 73].
3. You open the dBase attributes table for the Shapefile (.dbf) in Excel, and maintain the SAPBWKEY column [Page 74].
4. You load the copied Shapefile into the BI system [Page 75].

In this example scenario using the 0D_COUNTRY characteristic, the SAPBWKEY column is already maintained in the attributes table and corresponds with the SAP country keys in the master data table. If you maintain a Shapefile where the SAPBWKEY has not been maintained, or where the SAPBWKEY is filled with values that do not correspond to BI master data, you proceed as described in the steps above.

Result

You are now able to use the characteristic as a static geo-characteristic in the Business Explorer. Every user that works with a query containing this static geo-characteristic, is able to attach a map to the query and analyze the data on the map directly.

4.1.6.5.3.2.1.2.1 Creating a Local Copy of the Shape File

Use

You need a local copy of the shape file before you are able to maintain the SAPBWKEY column in the attributes table of the shape file.

Procedure

1. Use your file manager (Windows Explorer, for example) to localize the three files *cntry200.shp*, *cntry200.shx* and *cntry200.dbf* on the ESRI BI map data CD, and copy the files to the C:\SAPWorkDir directory, for example.
2. You must deactivate the *Write Protected* option before you are able to edit the files. (Select the files and choose the *Properties* option from the context menu (secondary mouse-click). Under *Attributes*, deactivate the *Write Protected* option).

If you do not have access to the ESRI BI map data CD, proceed as follows:

The files are already maintained in the BI Business Document Service (BDS). The following example explains how, for the characteristic 0D_COUNTRY in InfoCube 0D_SD_C0, you download these files from the BDS to your local directory.

1. Log on to the BI system and go to the InfoObject maintenance screen (transaction **RSD1**). This takes you to the *Edit InfoObjects: Start screen*.
2. In the InfoObject field, enter 0D_COUNTRY and choose *Display.* You reach the *Display Characteristic 0D_COUNTRY: Details* screen.
3. Choose the *Business Explorer* tab page. In the *BEx Map* area, 0D_COUNTRY is shown as a static geo-characteristic.
4. Choose *Display Shape files*. This takes you to the *Business Document Navigator* that already associates three shape files with this characteristic.
5. Open up the shape files completely in the *BI Metaobjects* tree.
6. Select the .dbf file BW_GIS_DBF and choose *Export Document*. This loads all the files to your local SAPWorkDirectory. (The system proposes the C:\SAPWorkDir directory as your SAPWorkDirectory).
7. Repeat the last step for the .shp (BW_GIS_SHP) and .shx (BW_GIS_SHX) files.

4.1.6.5.3.2.1.2.2 Downloading BI Master Data into a dBase File

Use

To maintain the SAPBWKEY column in the shapefile attribute table, you have to specify the corresponding BI country key for every row in the attribute table. As this information is

contained in the BI master data table, you have to download it into a local dBase file to compare it with the entries in the attribute table and maintain the SAPBWKEY.

Prerequisites

You have created a local working copy of the shapefile [Page 73].

Procedure

1. Log on to the BI system and go to the InfoObject maintenance screen (transaction **RSD1**). This takes you to the *Edit InfoObjects: Start screen*.
2. In the InfoObject field, enter 0D_COUNTRY and choose *Display.* The *Display Characteristic 0D_COUNTRY: Detail* dialog box appears.
3. Choose the *Business Explorer* tab page. In the *BEx Map* area, 0D_COUNTRY is displayed as a static geo-characteristic.
4. Choose *Geo Data Download (All).*
5. Accept the file name proposed by the system by choosing *Transfer.*

The proposed file name is made up of the technical name of the characteristic and the .dbf extension, therefore, in this case the file is called 0D_COUNTRY.DBF.

If the *Geo Data Download (All)* pushbutton is deactivated (gray), there is no master data for the InfoObject. If this is the case, download the texts for the InfoObject manually to get to the SAPBWKEY.

See also: Creating InfoObjects: Characteristics [Page 60], Tab Page: Master Data/Texts [Page 80]

Result

The status bar contains information on how much data has been transferred.

If you have not specified a directory for the file name, the file is saved in the local SAP work directory.

4.1.6.5.3.2.1.2.3 Maintaining the SAPBWKEY Column

Prerequisites

You have completed the following steps:

Created a local copy of the shapefile [Page 73]

Downloading BI master data into a dBase file [Page 73]

Integration

The SAPBWKEY is maintained in the dBase file with the suffix **.dbf**. This file contains the attributes table.

Procedure

1. Launch Microsoft Excel and choose *File* → *Open.*
2. From the dropdown box in the *Files of Type* field, choose *dBase Files (*.dbf).*
3. From the *C:\SAPWorkDir* directory, open the *cntry200.dbf* file. The attributes table from the shapefile is displayed in an Excel worksheet.
4. Repeat this procedure for the *0D_COUNTRY.DBF* file that you created in the step; Loading BI Master Data into the dBase File [Page 73]. This file shows you which values from the SAPBWKEY are used for which countries.
5. In the *0D_COUNTRY.DBF* file, use the short description (*0TXTSH* column) to compare the two tables.

   ESRI delivers an ESRI BI map data CD. This CD contains the SAPBWKEY (corresponding to the SAP country key) for the characteristic 0D_COUNTRY. This is why the *SAPBWKEY* column in the *cntry200.dbf* file is already filled with the correct values.

   Copy the SAPBWKEY manually to the attributes table in the shapefile

   – if you are using a different country key

   – if you are working with characteristics for which the *SAPBWKEY* column has not been defined, or is filled with invalid values

   If you are working with compounded characteristics, copy the complete SAPBWKEY, for example, for region *01* compounded with country *DE* copy the complete value *DE/01*.

   Do not under any circumstances change the sequence of the entries in the attributes table (for example, by sorting or deleting the rows!) If you were to change the sequence of the entries, the attributes table would no linger agree with the index and the geometric files.
6. When you have finished maintaining the SAPBWKEY column, save the attributes table in the shapefile, in this example, *cntry200.dbf*.

4.1.6.5.3.2.1.2.4 Uploading Edited Shapefiles into BI Systems

Prerequisites

You have completed the following steps:

Created a local copy of the shapefile [Page 73]

Downloaded BI master data into a dBase file [Page 73]

Maintained the SAPBWKEY column [Page 74]

Procedure

The last step is to attach the shapefile set (.shp, .shx, .dbf) to the InfoObject, by uploading it into the Business Document Service (BDS) on the BI server.

1. Log on to the BI system and go to the InfoObject maintenance screen (transaction **RSD1**). This takes you to the *Edit InfoObjects: Start screen*.
2. In the InfoObject field, specify 0D_COUNTRY and choose *Maintain.* This takes you to the *Change Characteristic 0D_COUNTRY: Detail* screen.
3. In the *Business Explorer* tab page, choose *Upload Shape Files.* The *Business Document Service: File Selection* dialog box appears.
4. Select the *cntry200.shp* file and choose *Open* The Business Document Service suggests entries for the file name, description, and so on, and allows you to enter key words that will make it easier for you to find the file in the BDS at a later date.
5. Choose *Continue*.
6. The system automatically asks you to upload the *cntry200.dbf* and *cntry200.shx* files for the shapefile.

Result

You have uploaded the edited shape file into the BI system. You can now use the characteristic in the Business Explorer. Every user that works with a query that contains the 0D_COUNTRY InfoObject, can now attach a map to the query and analyze the data on the map.

4.1.6.5.3.2.1.3 Geocoding

Purpose

To display dynamic geo-characteristics as points on a map, you have to determine the geographic co-ordinates for every master data object.

The master data table for dynamic geo-characteristics is, therefore, extended with a number of standard geo-attributes such as LONGITUDE and LATITUDE (see Static and Dynamic Geo-Characteristics [Page 67]).

Prerequisites

You have marked the geo-relevant characteristic as geo-relevant in the InfoObject maintenance. See the tab page: Business Explorer [Page 63]

To follow the example that explains the geocoding process, you must ensure that SAP DemoContent is active in your BI system.

Process Flow

Geocoding is implemented with ArcView GIS software from ESRI. ArcView GIS determines the geographical coordinates of BI data by identifying a column with geo-relevant characteristics in a reference Shapefile. To carry out this process, you have to load the BI master data table into a dBase file. The geographical coordinates are determined for every master data object. After you have done this, convert the dBase file with the determined geo-attributes into a CSV file (**c**omma-**s**eparated **v**alue file), which you can use for a master data upload into the BI master data table.

The following steps explain the process of geocoding dynamic geo-characteristics using the 0D_SOLD_TO characteristic (*Sold-to Party*) from the 0D_SD_C03 *Sales Overview Demo-Content InfoCube.*

1. You download BI master data into a dBase file [Page 77].
2. You execute the geocoding with ArcView GIS [Page 78] .
3. You convert dBase files into a CSV files [Page 79].
4. You schedule a master data upload for the CSV file.

   The system administrator is responsible for the master data upload.

Result

You are now able to use the characteristic as a dynamic geo-characteristic in the Business Explorer. Each user who works with a query that contains this dynamic geo-characteristic can now analyze data from a chart.

4.1.6.5.3.2.1.3.1 Downloading BI Master Data into a dBase File

Use

The first step in SAPBWKEY maintenance for dynamic geo-characteristics and their geocoding is to download the BI master data table into a dBase file.

Procedure

1. Log on to the BI system and go to the InfoObject maintenance screen (transaction **RSD1**). The *Edit InfoObjects: Start* dialog box appears.
2. In the InfoObject field, enter the name of the **dynamic** geo-characteristic that you want to geocode (in this example: 0D_SOLD_TO).
3. Choose *Display.* The *Display Characteristic 0D_SOLD_TO: Detail* dialog box appears.
4. Choose the *Business Explorer* tab page. In the *BEx Map* area, *0DSOLD_TO* is displayed as a *Dynamic Geo-Characteristic.*
5. Choose *Geo Data Download (All).*

   If you only want to maintain those entries that have been changed since the last attribute master data upload, choose *Geo Data Download (Delta).* The geo-data has to be downloaded in the delta version before you execute the realignment run for the InfoObject. Otherwise the delta information is lost.

6. The system asks you to select a geo-attribute that you want to include in the dBase file. The system only displays those attributes that were defined as geo-relevant. In this case, select both attributes: 0D_COUNTRY and 0D_REGION.
7. Choose *Transfer Selections*.
8. Transfer the file name suggested by the system and choose *Transfer*.

   The proposed file name is made up of the technical name of the characteristic and the .dbf extension. You can change the file name and create a directory. If you do not specify a path, the file is automatically saved in the SAP work directory.

Result

The status bar contains information on how much data has been transferred.

4.1.6.5.3.2.1.3.2 Geocoding Using ArcView GIS

Prerequisites

- You have installed the ESRI ArcView software on your system and requested the geographical data you need from ESRI, if this is not already on the data CD delivered with the software.
- You have completed the following step:

  Downloading BI master data into a dBase file [Page 77]

Use

Using geocoding, you enhance dynamic geo-characteristics from BI master data with the geographical attributes degrees of longitude and latitude.

Procedure

The following procedure is an example procedure that you can reconstruct using the demo contents. For further details on geocoding and on the ArcView functions, refer to ArcView documentation.

In ArcView GIS you can execute many commands easily from the context menu. To open the context menu, select an element and click on it with the secondary mouse button.

1. Open using *Programs* → *ArcGIS* *ArcCatalog*.
2. Under *Address Locators,* double-click on the entry *New Address Locator.*
3. In the *Create New Address Locator* window, select the entry *Single Field (File)* and click on OK.
4. In the *New: Single Field (File) Address Locator* window, enter the name of the service and the description, for example, **`Geocoding Service SoldTo`**. Under *Reference*

data, enter the path for the reference Shapefile, for example, *g_stat00.shp* and from the *Fields* dropdown menu, select the most appropriate entry, in this case, *SAPBWKEY.* Under *Output Fields*, activate the control box *X and Y Coordinates.*

In the navigation menu, the new service is displayed under *Address Locators.*

5. Open using *Programs → ArcGis ArcMap* and start with *A New, Empty Map* in the entry dialog. Choose *OK*.
6. In the standard toolbar, click on the *Add Data* symbol and add the corresponding dBase file, for example, *SoldTo.dbf* as a new table.

   The *Choose an address locator to use..* window is opened. All available services are displayed in this window.
7. Click *Add* and, in choose the *Address Locator entry in the Add Address Locator window under Search in:.* Select the service that you created in step four (in this example, *Geocoding Service SoldTo*) and click on *Add.*
8. In the *Choose an address locator to use..* window, select the service again, and click *OK.*

   The *Geocode Addresses* window is opened.
9. Under *Address Input Fields,* choose the appropriate entry, for example, *1_0D_Regio*. This is the field that tallies with the reference data. Under *Output Output Shapefile or feature class,* enter the path under which the result of the geocoding is to be saved. Choose *OK*.

   The data is geocoded.
10. After you have checked the statistics in the *Review/Rematch Addresses* window, click *Done.*

Result

The dynamic geo-characteristics for your master data have now been enhanced with additional geo-information in the form of the columns *X* (longitude) and *Y* (latitude). In ArcMap this information is displayed by points displayed in the right-hand side of the work area.

To check whether the result appears as you had planned, you can place the points on the relevant map. Proceed as follows:

1. Click on the *Add Data* symbol on the tab page.
2. Select the reference Shapefile that you used in step four, for example, *g_stat00.shp.*
3. Click *Add*.

   The map is displayed in the work area in a layer beneath the points.

4.1.6.5.3.2.1.3.3 Converting dBase Files into CSV Files

Prerequisites

You have completed the following steps:

Downloading BI master data into a dBase file [Page 77]

Geocoded using ArcView GIS [Page 78]

Integration

The result of the geocoding is the dBase file *Geocodierung_Result.dbf*. This file contains the BI master data enhanced with columns *X* and *Y*. Since the attribute table is stored in dBase file format, you must convert it into CSV (**c**omma **s**eparate **v**alue) format, which is executed by the BI Staging Engine. You can convert the table in Microsoft Excel.

Procedure

1. Launch Microsoft Excel and choose *File* → *Open...*
2. From the selection list in the field *Files of Type*, choose *dBase Files (*.dbf).*
3. Open the *Geocoding_Result.dbf* file. The attribute table with the geoattributes is displayed in Excel.
4. Choose *File* → *Save As...*
5. From the *Save as Type* selection list, choose *CSV (Comma Delimited).*
6. Save the table.

Result

You have converted the dBase file into a CSV file with the geoattribute for the dynamic geocharacteristic 0D_SOLD_TO. You system administrator can now schedule a master data upload.

When you upload the CSV file, you have to map the values in column X to the attribute 0LONGITUDE, and the values in column Y to the attribute 0LATITUDE.

4.1.6.5.3.3 Tab Page: Master Data/Texts

Use

On this tab page you specify whether the characteristic can have master data and/or texts.

Structure

With Master Data

Here you specify whether the characteristic has master data. You have to create a master data table for a characteristic if you want the characteristic to have attributes.

When you select this option, the system generates a master data table. You branch to the relevant maintenance dialog by choosing Maintain Master Data [Page 115].

The master data table can have a time-dependent and a time-independent part. See also Master Data Types: Attributes, Texts and Hierarchies [Page 114]

In attribute maintenance, you specify whether an attribute is time dependent or time independent.

With Texts

Here you specify whether the characteristic has texts.

If you want to use texts with a characteristic, you have to select at least one text. The *short text* (20 characters) option is set by default but you can also choose medium-length texts (40 characters) or long texts (60 characters).

Language-Dependent Texts

You can choose whether you want the texts in the text table to be language dependent. If you specify that you want texts to be language dependent, the language is a key field in the text table. If you specify that you do not want the texts to be language dependent, there is no language field in the text table.

Some BI Content characteristics, for example customer (0CUSTOMER), do not need to be language dependent.

Time-Dependent Texts

If you want texts to be time dependent (the date is included in the key of the text table), you make the appropriate settings here. See also: Using Master Data and Characteristics That Bear Master Data [Page 113]

Master Data Maintenance with Authorization Check

If you set this indicator, you can use authorizations to protect the master data and texts for this characteristic from being maintained at single-record level. If you activate this option, for each key field of the master data table, you can enter the characteristic values for which the user has authorization. You do this in the profile generator in role maintenance using authorization object S_TABU_LIN.

See Authorizations for Master Data [External].

If you do not set this indicator, you can only allow access to or lock the entire maintenance of master data (for all characteristic values).

DataStore Object for Checking Characteristic Values

If you create a DataStore object for checking the characteristic values of a characteristic, in the update rules and transfer rules, the valid values for the characteristic are determined from the DataStore object not the master data. The DataStore object must contain the characteristic itself and all the fields in the compound as key figures.

See Checking for Referential Integrity [Page 279].

Characteristic Is ....

InfoSource:

If you want to use a characteristic as an InfoSource (by means of direct update), you have to assign an application component to the characteristic. The system displays the characteristic in the InfoSource tree in the Data Warehousing Workbench. You can assign DataSources and source systems to the characteristic. You can also load master data, texts, and hierarchies for the characteristic.

In the following cases you **cannot** use an InfoObject as an InfoSource with direct update:

- The characteristic is characteristic 0SOURSYSTEM (source system ID).
- The characteristic has no master data, texts or hierarchies – there is no point in loading data for the characteristic.

- The characteristic is not a characteristic but a unit or key figure.

For more information, see InfoSource Types [Page 272].

If you want to generate an export DataSource for a characteristic, the characteristic has to be an InfoSource with direct update; it has to be assigned to an application component.

InfoProvider:

This indicator specifies whether the characteristic is an InfoProvider.

If you want to use a characteristic as an InfoProvider, you have to assign an InfoArea to the characteristic. The system displays the characteristic in the InfoProvider tree in the Data Warehousing Workbench. You can use the characteristic as an InfoProvider in reporting.

You can only use a characteristic as an InfoProvider if the characteristic contains texts or master data.

You can define queries for the characteristic (more precisely, for the master data of the characteristic) if you are using a characteristic as an InfoProvider. In this case, on the *Attributes* tab page, you can switch on dual-level navigation attributes (navigation attributes for navigation attributes) for this characteristic in its role as an InfoProvider.

For more information see InfoObjects as InfoProviders [Page 165].

Export DataSource:

If you set this indicator, you can extract the master data, texts, and hierarchies of the characteristic into other BI systems. See also Data Mart Interface [External].

Master Data Access

You have three options for accessing the master data at query runtime:

1. Standard: The system displays the values in the master data table for the characteristic. This is the default setting.
2. Own implementation: You can define an ABAP class to implement the access to master data yourself. You need to implement interface IF_RSMD_RS_ACCESS. You need to be proficient in ABAP OO. An example of this is the time characteristic 0FISCYEAR that is delivered with BI Content.
3. Direct: If the characteristic is selected as an InfoProvider, you can access the data in a source system using direct access. If you choose this option, you have to use a data transfer process to connect the characteristic to the required DataSource and you have to assign the characteristic to a source system.

We recommend that you use the standard default setting. If you have special requirements with regard to reading master data, you can use a customer-defined implementation.

We recommend that you do not use direct access to master data in performance-critical scenarios.

4.1.6.5.3.4 Tab Page: Hierarchy

Use

If you want to create a hierarchy, or upload an existing hierarchy from a source system, you have to set the *with hierarchy* indicator. The system generates a hierarchy table with hierarchical relationships for the characteristic.

You are able to determine the following properties for the hierarchy:

- Whether or not you want to create hierarchy versions [External] for a hierarchy.
- Whether you want the entire hierarchy or just the hierarchy structure to be time-dependent [External].
- Whether you want to allow the use of hierarchy intervals [External].
- Whether you want to activate the sign reversal [External] function for nodes.
- The characteristics that are permitted in the hierarchy nodes: If you want to use the PSA to load your hierarchy, you must select InfoObjects for the hierarchy basic characteristic that you want to upload as well. All the characteristics you select here are included in the communication structure for hierarchy nodes, together with the characteristics compounded to them. For hierarchies that are loaded using IDocs [External], it is a good idea to also select the permitted InfoObjects. This makes maintenance of the hierarchy more transparent, because only valid characteristics are available for selection.

  If you do not select an InfoObject here, only text nodes are permitted as nodes that can be posted to in hierarchies.

See also:

Hierarchies [External]

Using Master Data and Master Data-Bearing Characteristics [Page 113]

4.1.6.5.3.5 Tab Page: Attributes

Use

On this tab page, you determine whether or not the characteristic has attributes or navigation attributes, and if so, which properties they have.

This tab page is only available if you have set the *With Master Data* indicator on the *Master Data/Texts* tab page.

Structure

Attributes are InfoObjects that exist already, and that are assigned logically to the new characteristic. There are the following ways to maintain attributes for a characteristic:

- Choose attributes from the *Attributes of the Assigned DataSources* list.
- Use F4 Help for the fields that are ready for input in the *Attributes of the Characteristic* list, to display all the InfoObjects. Choose the attributes that you need.
- In the *Attributes* list, specify directly in the fields that are ready for input the name of an InfoObject that you want to use as an attribute. If the InfoObject that you want to use does not yet exist, you have the option of creating a new InfoObject at this point. Any

new InfoObjects that you create are inactive. They are activated when the existing InfoObject is activated.

Properties

Choose *Detail / Navigation Attribute* to display the detailed view. In the detail view, you set the following:

Time Dependency

You can decide for each attribute individually whether it is to be time dependent. If only one attribute is time dependent, a master data table is created. However, there can still be attributes for this characteristic that are not time dependent.

All time-dependent attributes are in one table, meaning that they all have the same time-dependency, and all time-constant attributes are in one table.

Characteristic: Business process

Table /BI0/PABCPROCESS - for time-constant attributes

Characteristic: Business process	**Attribute: Cost center responsible**
Characteristic value: 1010	Attribute value: Jones

Table /BI0/QABCPROCESS - for time-dependent attributes

Business Process	**Valid From**	**Valid To**	**Company Code**
Characteristic value: 1010	01.01.2000	01.06.2000	Attribute value: A
	02.06.2000	01.10.2000	Attribute value: B

A view, /BI0/MABCPROCESS, connects these two tables:

Business Process	**Valid From**	**Valid To**	**Company Code**	**Cost Center Responsible**
1010	01.01.2000	01.06.2000	A	Jones
	02.06.2000	01.10.2000	B	Jones

In master data updates, you can either load time-dependent and time-constant data individually, or together.

Sequence of Attributes in Input Help

You can determine the sequence in which the attributes for a characteristic are displayed in the input help. There are the following values for this setting:

- **00:** The attribute is not displayed in the input help
- **01:** The attribute appears in the first position (far left) in the input help
- **02:** The attribute appears in the second position in the input help.
- **03:** .....

Altogether, only 40 fields are permitted in the input help. In addition to the attributes, the characteristic itself, its texts, and the compounded characteristics are generated in the input help. The total number of fields cannot be greater than 40.

Navigation Attribute

You can switch navigation attributes on and off, and specify their descriptions and short texts. These texts for navigation attributes can also be supplied by the basic InfoObject. If the text of the characteristic changes, the texts for the navigation attribute are adjusted automatically. This process requires very little maintenance and translation resources.

When you are defining and executing queries, it is not possible to use the texts to distinguish between navigation attributes and characteristics.

As soon as a characteristic appears in duplicate (as a characteristic and as a navigation attribute) in an InfoProvider, you must give the navigation attribute a different name. For example, you could call the characteristic *Cost Center*, and call the navigation attribute *Person Responsible for the Cost Center*.

See also the topic Elimination of Internal Business Volume [External]: The characteristic pair *Sent Cost Center* and *Received Cost Center* has the same reference characteristic and has to be differentiated by the text.

Authorization Relevance

You can mark navigation attributes as authorization relevant independently of the assigned basic characteristics.

Navigation Attributes for InfoProviders

For characteristics that are identified as InfoProviders, you can maintain two-level navigation attributes (that is, navigation attributes of navigation attributes) using *Navigation Attribute InfoProviders.* This is used for master data reporting on the characteristic. See also InfoObject as InfoProvider [Page 165].

It has no effect on characteristics used in other InfoProviders. In other words, if you use this characteristic in an InfoCube, the 2-level navigation attributes are **not** available for reporting on this InfoCube.

4.1.6.5.3.6 Tab Page: Compounding

Use

In this tab page, you determine whether you want to compound the characteristic to other InfoObjects. You sometimes need to compound InfoObjects in order to map the data model. Some InfoObjects cannot be defined uniquely without compounding.

For example, if storage location A for plant B is not the same as storage location A for plant C, you can only evaluate the characteristic *Storage Location* in connection with *Plant*. In this case, compound characteristic *Storage Location* to *Plant*, so that the characteristic is unique.

One particular option with compounding is the possibility of compounding characteristics to the source system ID [Page 87]. You can do this by setting the *Master data is valid locally for*

the source system indicator. You may need to do this if there are identical characteristic values for the same characteristic in different source systems, but these values indicate different objects.

Using compounded InfoObjects extensively, particularly if you include a lot of InfoObjects in compounding, can influence performance. Do not try to display hierarchical links through compounding. Use hierarchies [External] instead.

A maximum of 13 characteristics can be compounded for an InfoObject. Note that characteristic values can also have a maximum of 60 characters. This includes the concatenated value, meaning the total length of the characteristic in compounding plus the length of the characteristic itself.

Reference InfoObjects

If an InfoObject has a reference InfoObject, it has its technical properties:

- For characteristics these are the data type and length as well as the master data (attributes, texts and hierarchies). The characteristic itself also has the operational semantics.
- For key figures these are the key figure type, data type and the definition of the currency and unit of measure. The referencing key figure can have another aggregation.

These properties can only be maintained with the reference InfoObject.

Several InfoObjects can use the same reference InfoObject. InfoObjects of this type automatically have the same technical properties and master data.

The operational semantics, that is the properties such as description, display, text selection, relevance to authorization, person responsible, constant, and attribute exclusively, are also maintained with characteristics that are based on one reference characteristic.

The characteristic *Sold-to Party* is based on the reference characteristic *Customer* and, therefore, has the same values, attributes, and texts.

More than one characteristic can have the same reference characteristic: The characteristics *Sending Cost Center* and *Receiving Cost Center* both have the reference characteristic *Cost Center*.

See the documentation on eliminating internal business volume [External].

Characteristic Constants

When you assign a constant, a fixed value is assigned to a characteristic. The characteristic then exists on the database (for example, verifications), but it is not visible in the query.

The *Storage Location* characteristic is compounded with the *Plant* characteristic. If only one plant is ever run within the application, a constant can be assigned to the plant. The verification for the storage-location master table runs correctly with this value for the plant.

Special case:
If you want to assign the SPACE constant (type CHAR) or 00..0 (type NUMC) to the characteristic, type # in the first position.

4.1.6.5.3.7 Characteristic Compounding with Source System ID

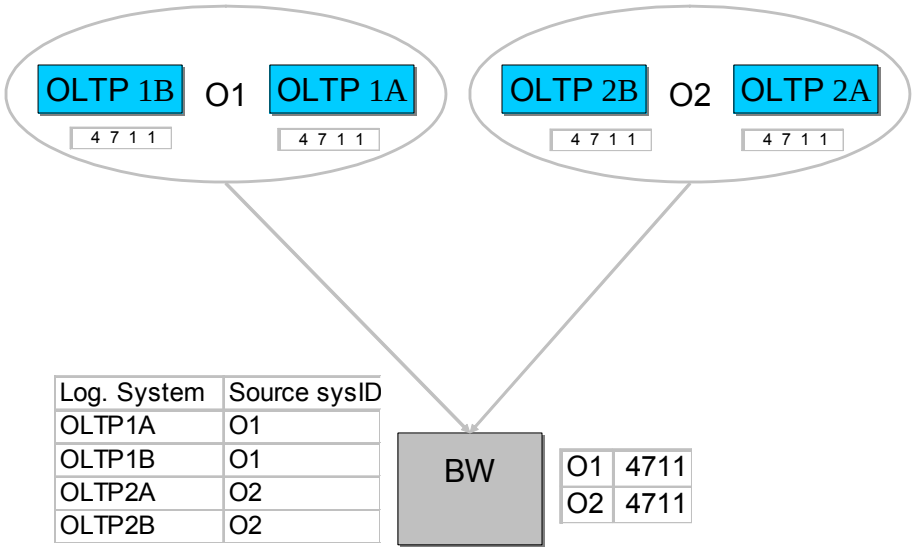
Use

If there are identical characteristic values describing different objects for the same characteristic in various source systems, you have to convert the values in such a way in SAP BW so as to make them unique.

For example, the same customer number may describe different customers in different source systems.

You can carry out conversion in the transfer rules for this source system or in the transfer routine for the characteristic.

If work involved in conversion is too great, you can compound the characteristic to the InfoObject *Source System ID* (0SOURSYSTEM). This means it is automatically filled with master data. The *Source System ID* is a 2-character identifier for a source system or a group of source systems in BW. The source system ID is updated with the ID of the source systems that provides the data. Assigning the same ID to more than one source system creates a group of source systems. The master data is unique within each group of source systems.



You already have 10 source systems within which the master data is unique. Five new source systems are now added, resulting in overlapping. You can now assign the 10 existing source systems to ID 'OL' (with text 'Old Systems') and

the 5 new systems to ID 'NE' (Text: 'New Systems'). Note: You now need to reload the data.

If you use characteristic *Source System ID*, you have to assign an ID to each source system.

If you did not assign an ID to each source system, an error will occur when you load master data for the characteristics that use the *Source System ID* as attribute or in the compounding. This is because, in data transfers, the source system to source system ID assignment is used to determine which value is updated for the characteristic *Source System ID*.

Master Data that is Local in the Source System (or Group of Source Systems)

If you have master data that is only unique locally for the source system in SAP BW, you can compound the relevant characteristics to the *Source System ID* characteristic. In this way, you can separate identical characteristic values that refer to different objects in different systems.

Data transfers from one BW system into another BW system are an exception, that is, where this 1:1 assignment does not apply. See also the section ***Exception Scenario: Data Mart*** in Assigning a Source System to a Source System ID [Page 88].

RRI (Report-Report-Interface) and Drag & Relate

Prior to Release SAP BW 3.0, the *Source System ID* characteristic was also used to return to the source system. Because this is not unique, however, the source system (0LOGSYS) is used as attribute starting with Release SAP BW 3.0 since with this release more than one source system can be grouped to one source system ID.

Characteristics that are to be traced in your original system using the RRI (Report-Report-Interface) or Drag & Relate should have characteristic 0LOGSYS as attribute.

When you integrate your BW system into SAP Enterprise Portal, the *Source System* characteristic is used to define the logical system of the business objects corresponding to the characteristic values. In this system then, functions specified using Drag& Relate are called (for example the detail display of an order or a cost center). Every characteristic of the Business Content that corresponds to a business object has characteristic *Source System* as attribute.

If you assign more than one source system to a source system ID, you can define one system of this group as default system. This system is then used in the Report-Report-Interface and in Drag & Relate for the return jump. This default system is only used if the origin of the data was not yet uniquely defined by characteristic 0LOGSYS.

Deleting and Removing a Source System ID

You can only delete the assignment to a source system ID if it is no longer used in the master or transaction data. Use the *Release IDs that are not in use* function here.

4.1.6.5.3.7.1 Assigning a Source System to a Source System ID

Use

Assigning a source system to a source system ID is necessary if, for example, you want to compound a characteristic to the InfoObject ‘Source System ID’ [Page 87].

When data is transferred, the **source system to source system ID** assignment is used to determine which value is updated for the source system ID characteristic.

The source system ID indicates the source system from which data is delivered.

Procedure

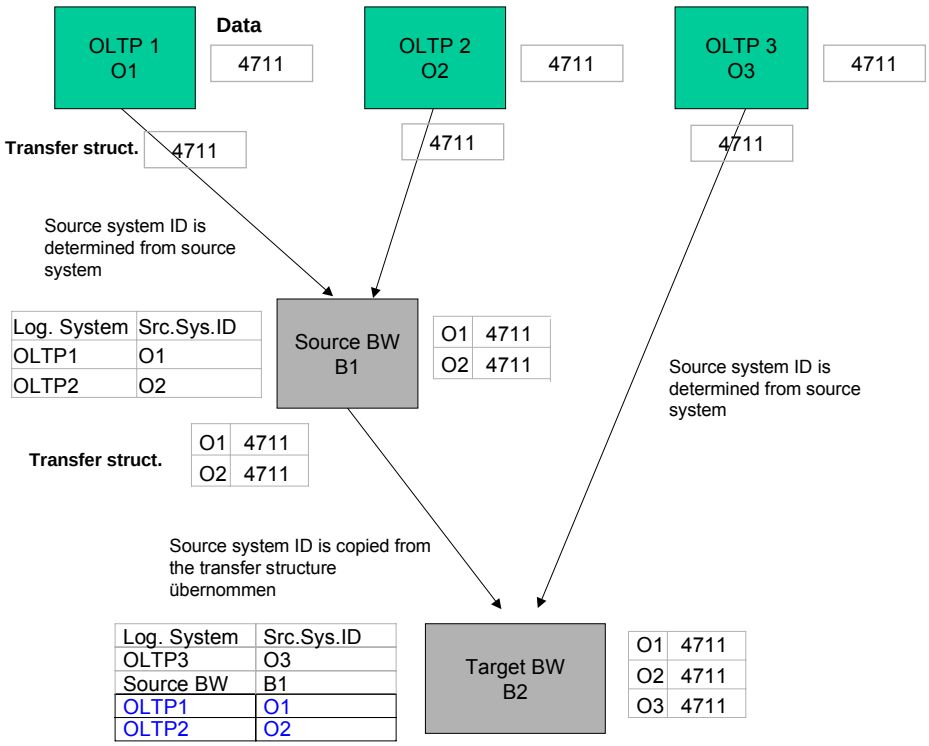
1. In the Data Warehousing Workbench, choose *Tools* → *Assignment Source System to Source System ID* from the main menu.
2. Choose *Suggest Source System IDs.*
3. Save your entries.

The source system ID for a source system can be changed if it is no longer being used in the master or transaction data. To do this, use the function *Release IDs that are not in use* in maintenance for source system ID assignment.

Exception Scenario: Data Mart

Data transfers from one BW system (**source BW**) into another BW system (**target BW**) are cases where this 1:1 assignment does not apply. The system ID for the source BI cannot be used here, since various objects that have been differentiated between in the source BI by compounding with the source system ID would otherwise collapse.

When you transfer data from the source BI to the target BI, the source system IDs are copied from the source BI. If these IDs are not yet recognized in the target BI, then you have to create them. It is possible to create source system IDs for logical systems that are not used as BI source systems.



Procedure

1. In the main menu of the Data Warehousing Workbench, choose *Tools → Assignment Source System to Source System ID.*
2. Choose *Create*.
3. Enter the *logical system name* and a *description*, and confirm you entries (in this example the name would be OLTP1 or OLTP2).
4. In the *Source System ID* column enter the ID name that you also entered in BW1 for the corresponding source system. (In this example it would be ID 01 or ID 02).
5. Save your entries.

4.1.6.5.3.8 Navigation Attribute

Use

Characteristic attributes can be converted into navigation attributes. They can be selected in the query in exactly the same way as the characteristics for an InfoCube. In this case, a new edge/dimension is added to the InfoCube. During the data selection for the query, the data manager connects the InfoProvider and the master data table (‘join’) in order to fill the Query.

Costs of the cost center drilled down by person responsible:

You use the attribute ‘Cost Center Manager’ for the characteristic ‘Cost Center’. If you want to navigate in the query using the cost center manager, you have to create the attribute ‘Cost Center Manager’ as a navigation attribute, and flag it as a navigation characteristic in the InfoProvider.

When executing the query there is no difference between navigation attributes and the characteristics for an InfoCube. All navigation functions in the OLAP processor are also possible for navigation attributes.

Extensive use of navigation attributes leads to a large number of tables in the connection (‘join’) during selection and can impede the performance of the following actions:

- Deletion and creation of navigation attributes (construction of attribute SID tables)
- Change of time-dependency of navigation attributes (construction of attribute SID tables)
- Loading master data (adjustment of attribute SID tables)
- Call up of input help for a navigation attribute
- Execution of queries

Therefore, only make those attributes into navigation attributes that you really need for reporting.

See also Performance of Navigation Attributes in Queries and Input Help [Page 92].

See also:

Create Navigation Attributes [Page 91]

4.1.6.5.3.8.1 Creating Navigation Attributes

Prerequisites

You are in InfoObject maintenance and have selected the tab page *Attributes*.

Procedure

1. Specify the technical name of the characteristic that you want to use as a navigation attribute, or create a new attribute by choosing *Create*. You can also directly transfer proposed attributes of the InfoSource.

   In order to use the characteristic as a navigation attribute, make sure the InfoObject is first assigned as an attribute, and that the option *Attribute Only* is not activated for the characteristic on the *General* tab page.

2. By clicking on the symbol *Navigation Attribute On/Off* in the relevant column, you can define an attribute as a navigation attribute.
3. When you set the indicator as Authorization Relevant the navigation attribute is checked upon executing an authorization query.
4. Choose the *Characteristic Texts* indicator, or specify a name in the *Navigation Attribute Description* field.

   If you turn a characteristic attribute into a navigation attribute, you can assign a text to the navigation attribute to distinguish it from a normal characteristic in reporting.

Result

You have created a characteristic as a navigation attribute for your superior characteristic.

Further Steps to Take

You must activate the created navigation attributes in the InfoProvider maintenance. The default is initially set to *Inactive* so as not to implicitly include more attributes than are necessary in the InfoCube.

Navigation attributes can affect performance. See also Performance of Navigation Attributes in Queries and Input Help [Page 92].

Note: You can create or activate navigation attributes in the InfoCube at any time. Once an attribute has been activated, you can only deactivate it if it is not used in aggregates [Page 666].

In addition, you must record your navigation attributes in queries so that they are included in reporting.

4.1.6.5.3.8.2 Performance of Navigation Attributes in Queries and Value Help

From a system performance point of view, you should model an object on a characteristic rather than on a navigation attribute, because:

- In the enhanced star schema of an InfoCube, navigation attributes lie one join further out than characteristics. This means that a query with a navigation attribute has to run an additional join (compared with a query with the same object as a characteristic) in order to arrive at the values. This is also true for DataStore objects.
- For the same reason, in some situations, restrictions for particular values in the navigation attribute (values that have been defined in the query) are not taken into account by the database optimizer when it creates run schedules. This can result in inefficient run schedules, particularly if the restrictions are very selective. In most cases, you can solve this problem by indexing the navigation attribute in the corresponding master data tables (see below.)
- If a navigation attribute is used in an aggregate, the aggregate has to be adjusted using a change run as soon as new values are loaded for the navigation attribute (when master data for the characteristic belonging to the navigation attribute is loaded). This change run is usually one of the processes critical to the system performance of a productive BI system. This is why avoiding using navigation attributes, or not using navigation attributes in aggregates, you can improve the performance of this process. On the other hand, not using navigation attributes in aggregates can lead to poor query response times. The data modeler needs to find the right balance.

Additional Indexing

It is sometimes appropriate to manually create additional indexes for master data tables, to improve system performance for queries with navigation attributes. A typical scenario would be if there were performance problems during the selection of characteristic values, for example:

- In BEx queries containing navigation attributes, where the corresponding master data table is large (more than 20,000 entries), there is usually a restriction placed on the navigation attributes.
- In the input help for this type of navigation attribute.

Example

You want to improve the performance of navigation attribute A in characteristic C. You have restricted navigation attribute A to certain values. If A is time-independent, you need to refer to the X table of C (/BI0/XC or /BIC/XC). If A is time-dependent, you need to refer to the Y table of C (/BI0/YC or /BIC/YC). This table contains a column S__A (A = Navigation attribute). Using the ABAP dictionary, for example, you need to create an additional database index for this column: *SAP Easy Access* → *Tools* → *ABAP Workbench* → *Development* → *Dictionary*.

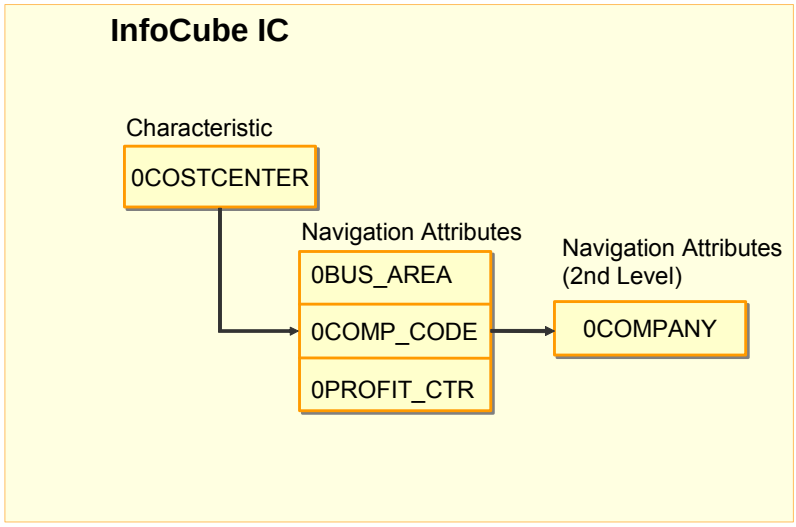
You must verify whether the index that you have created has actually improved performance. If there is no perceivable improvement, you must delete the index, as maintaining defunct indexes can lead to poor system performance when data is loaded (in this case master data) and has an impact on the change run.

4.1.6.5.3.8.3 Transitive Attributes as Navigation Attributes

Use

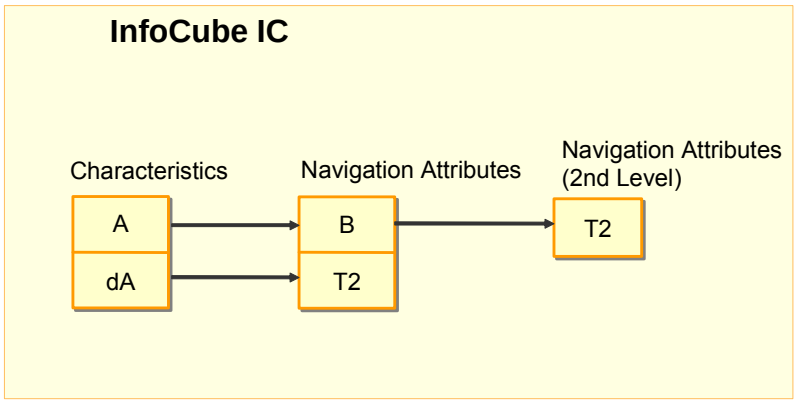
If a characteristic was included in an InfoCube as a navigation attribute, it can be used for navigating in queries. This characteristic can itself have further navigation attributes, called transitive attributes. These attributes are not automatically available for navigation in the query. As described in this procedure, they must be switched on.

An InfoCube contains InfoObject 0COSTCENTER (cost center). This InfoObject has navigation attribute 0COMP_CODE (company code). This characteristic in turn has navigation attribute 0COMPANY (company for the company code). In this case 0COMPANY is a transitive attribute that you can switch on as navigation attribute.



Procedure

In the following procedure, we assume a simple scenario with InfoCube IC containing characteristic A, with navigation attribute B and transitive navigation attribute T2, which does not exist in InfoCube IC as a characteristic. You want to display navigation attribute T2 in the query.



Creating Characteristics

Create a new characteristic dA (denormalized A) which has the transitive attributes requested in the query as navigation attributes (for example T2) and which has the same technical settings for the key field as characteristic A.

After creating and saving characteristic dA, go to transaction SE16, select the entry for this characteristic from table RSDCHA (CHANM = <characteristic name> and OBJVERS = 'M') and set field CHANAV to **2** and field CHASEL to **4**. This renders characteristic dA invisible in queries. This is not technically necessary, but improves readability in the query definition since the characteristic does not appear here.

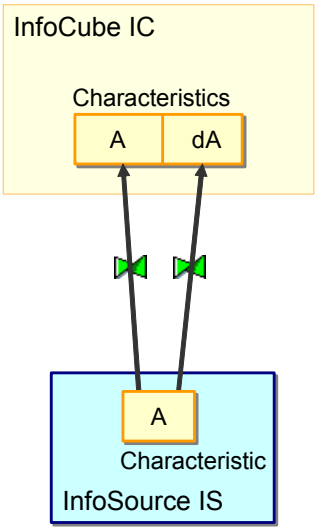
Start transaction RSD1 (InfoObject maintenance) again and activate the characteristic.

Including Characteristics in the InfoCube

Include characteristic dA in InfoCube IC. Switch on its navigation attribute T2. The transitive navigation attributes T2 are now available in the query.

Modifying Transformation Rules

Now modify the transformation rules for InfoCube IC so that the newly included characteristic dA is calculated in exactly the same way as the existing characteristic A. The values of A and dA in the InfoCube must be identical.



Creating InfoSources

Create a new InfoSource. Assign the DataSource of characteristic A to the InfoSource.

Loading Data

Technical explanation of the load process:

The DataSource of characteristic A must define the master data table of characteristic A as well as of characteristic dA. In this example the DataSource delivers key field A and attribute B. A and B must be updated to the master data table of characteristic A.

A is also updated to the master data table of dA (namely in field dA) and B is only used to determine transitive attribute T2, which is read from the updated master data table of characteristic B and written to the master data table of characteristic dA.

Since the values of attribute T2 are copied to the master data table of characteristic dA, this results in the following dependency, which must be taken into consideration during modeling:

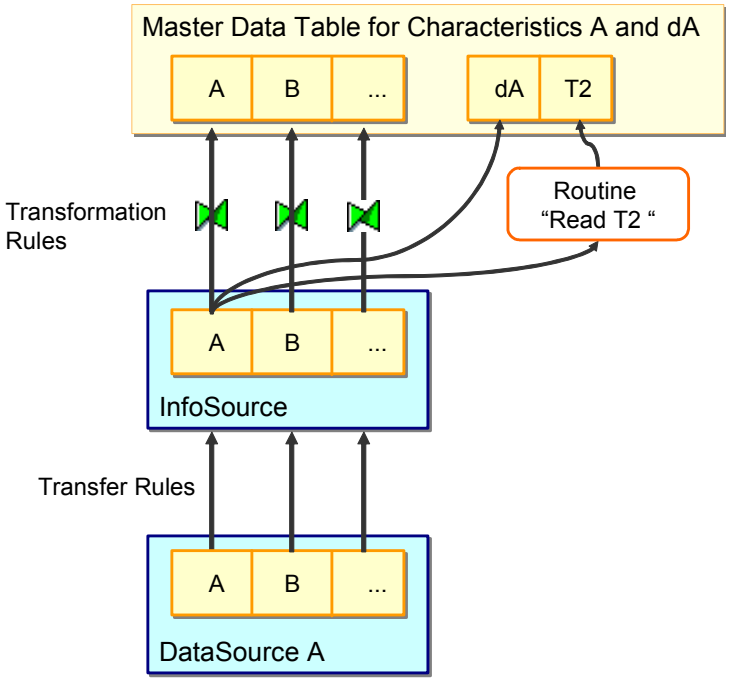
If a record of characteristic A changes, it is transferred from the source system when it is uploaded into the BI system. If a record of characteristic B changes it is transferred from the source system when it is uploaded into the BI system. However, since attribute T2 of characteristic B is read and copied when characteristic A is uploaded, a data record of characteristic A might not be transferred to the BÍ system during a delta upload of characteristic A because it has not changed. But the transitive dependent attribute T2 might have changed for this record only but the attribute would not be updated for dA.

The structure of a scenario for loading data depends on whether or not the extractor of DataSource A is delta enabled.

Loading process:

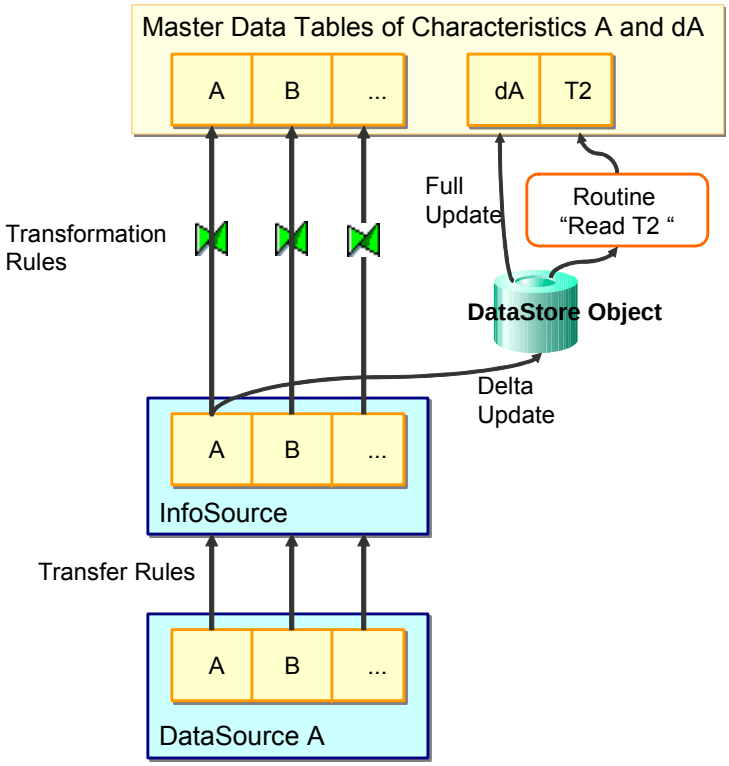
1. Scenario for non-delta-enabled extractor

If the extractor for DataSource A is not delta enabled, the data is updated to the two different InfoProviders (master data table of characteristics A and dA) using an InfoSource and two different transformation rules.



2. Scenario for delta-enabled extractor

If it is a delta-enabled extractor, a DataStore object from which you can always execute a full update in the master data table of characteristic dA is used. With this solution the data is also updated to two different InfoProviders (master data table of characteristic A and new DataStore object which has the same structure as characteristic A) in a delta update using a new InfoSource and two different transformation rules. Transformation rules from the DataStore object are also used to write the master data table of characteristic dA with a full update.



For both solutions, the transformation rules in the InfoProvider *master data table of characteristic dA* must cause attribute T2 to be read. For complicated scenarios in which you read from several levels, function modules will be retrieved that execute this service.

It is better for the coding for reading the transitive attributes (in the transformation rules) if you include the attributes to be read in the InfoSource right from the beginning. This means that you only have transformation rules that perform one-to-one mapping. The additional attributes that are included in the InfoSource are not filled in the transfer rules. They are only computed in the transformation rules in a start routine, which must be created. The advantage of this is that the coding for reading the attributes (which can be quite complex) is stored in one place in the transformation rules.

In both cases the order at load time must be adhered to and must be implemented either organizationally or using a process chain. It is essential that the master data to be read (in our case the master data of characteristic B) already exists in the master data tables in the system when the data providing the DataSource of characteristic A is loaded.

Change the master data from characteristic B so that it is also visible with the next load into A / dA.

4.1.6.5.3.9 Conversion Routines in the BI System

Use

Conversion routines are used in the BI system so that the characteristic values (key) of an InfoObject can be displayed or used in a different format to how they are stored in the database. They can also be stored in the database in a different format to how they are in their original form, and supposedly different values can be consolidated into one.

Conversion routines that are often implemented in the BI system are now described.

Integration

In the BI system, conversion routines essentially serve to simplify the input of characteristic values for a query runtime. For example with cost center 1000, the long value with left-hand zeros 0000001000 (from the database) is not to be entered, but just 1000. Conversion routines are therefore linked to characteristics (InfoObjects) and can be used by them.

Conversion routines can also be set with data loading. At the DataSource there are two conversion routines: one that is entered in the SAP source system and entered in the BI system at replication, and one that is defined in the BI system or was already defined for BI Content DataSources. In the DataSource maintenance you can define if the data is delivered in external or internal format, or if the format should be checked. The conversion routine from the source system is hidden there.

The conversion routine from the source system is used in the InfoPackage in the value help. The conversion routine in the BI system is checked upon loading (OUTPUT & INPUT), executed (INPUT) or ignored (in this case, when the DataSource is checked there is a warning if a conversion routine is nevertheless entered), depending on the setting made in the field. It is also used for the display (OUTPUT) and maintenance (INPUT) of the data in the PSA...

In many cases it is desirable to store the conversion routines of these fields in the corresponding InfoObject on the BI system side too. When the fields of the DataSource are assigned to the InfoObjects, a conversion routine is assigned by default in the transformation rules. You can choose whether or not to execute this conversion routine. Conversion routines PERI5, PERI6 and PERI7 are not executed automatically since these conversions are only performed when data is extracted to the BI system.

When loading data you now have to consider that when extracting from SAP source systems the data is already in the internal format and is not converted. When loading flat files and when loading using a BAPI or DB Connect, the conversion routine displayed signifies that an INPUT conversion is executed before writing to the PSA. For example, the date field is delivered from a flat file in the external format,10.04.2003’. This field can be converted in the transformation rules to internal format '20030410’ according to a conversion routine.

A special logic is used in the following cases: For numeric fields, a number format transformation is performed if needed (if no conversion routine is specified). For currencies, a currency conversion is also performed (if no conversion routine is specified). If required, a standard transformation is performed fort he date and time (according to the user settings). A more flexible user-independent date conversion is provided by conversion routine RSDAT.

Conversion routines ALPHA, NUMCV, and GJAHR check whether data exists in the correct internal format before it is updated. For more on this see the extensive documentation in the BI system in the transaction for converting to conforming internal values (transaction RSMDCNVEXIT). If the data is not in the correct internal form an error message is issued.

BI Content objects are delivered with conversion routines if they are also used by the DataSource in the source system. The external presentation is then the same in both

systems. The name of the used conversion routines of the DataSource fields is transferred to the BI system when the DataSources are replicated from the SAP source systems.

Features

A conversion occurs according to the data type of the field when changing the content of a field from the display format into the SAP-internal format and vice versa, as well as for output using the ABAP WRITE instruction. The same is true for output using a BI system query.

If this standard conversion is unsuitable you can override it by specifying a conversion routine in the underlying domains. You do this in the BI system by specifying a conversion routine in InfoObject maintenance in the General Tab Page [Page 61].

See Defining Conversion Routines [External] for more technical details.

4.1.6.5.3.9.1 ALPHA Conversion Routine

Use

The ALPHA conversion is used in the BI system for each presetting for character characteristics. The ALPHA conversion routine is registered automatically when a characteristic is created. If you do not want to use this routine, you have to remove it manually.

The ALPHA conversion routine is used, for example, with account numbers or document numbers.

Features

When converting from an external into an internal format this checks whether the entry in the INPUT field is wholly numerical, whether it consists of digits only, possibly with blank spaces before and/or after. If yes, the sequence of digits is copied to the OUTPUT field, right-aligned, and the space on the left is filled with zeros (‘0’). Otherwise the sequence of digits is copied to the output field from left to right and the space to the right remains blank.

For conversions from an internal to an external format (function module CONVERSION_EXIT_ALPHA_OUTPUT) the process is reversed. Blank characters on the left-hand side are omitted from the output.

Example

Input and output fields are each 8 characters long. A conversion from the external to the internal format takes place:

1. '1234 ' → '00001234'
2. 'ABCD ' → 'ABCD '

4.1.6.5.3.9.2 BUCAT Conversion Routine

Use

The BUCAT conversion routine converts the internal presentation of the budget type (0BUD_CAT) into the external presentation (0BUD_CAT_EX), using the active entries in the master data table for the budget type InfoObject (0BUD_CAT).

Example

Conversion from the external into the internal format:

'1' → 'IM000003'

4.1.6.5.3.9.3 EAN11 Conversion Routine

Use

The EAN11 conversion routine is used for European Article Numbers (EAN) and the American Universal Product Code (UPC).

Features

It converts the external presentation, according to settings in transaction W4ES (in the ERP system), into the internal SAP presentation. In the SAP system, left-hand zeros are not saved as, according to EAN standards, these are not required. For example, the EAN ‘123’ is the same as the EAN ‘00123’. As such, the left-hand zeros are dispensed with.

UPC-E code short forms are converted into the long form.

The EAN11 conversion routine formats the internal presentation of each EAN type, according to settings in transaction W4ES, for output. This ensures that the internal presentation does have left-hand zeros, or that UPC codes are converted to the short form.

4.1.6.5.3.9.4 GJAHR Conversion Routine

Use

Conversion routine GJAHR is used when entering the business year in order to allow an abbreviated, two-digit entry. A business year has four digits in the internal format.

Functions

When converting from an external into an internal format this checks whether the entry in the INPUT field is wholly numerical, whether it consists of digits only, possibly with blank spaces before and/or after.

1. If a two-digit sequence of numbers is entered then these are put in the third and fourth spaces of the OUTPUT field. The left-hand spaces are filled with 19 or 20 according to the following rule:
    - Two-digit sequence < 50. Fill from left with 20.
    - Two-digit sequence >= 50. Fill from left with 19.

2. A sequence that does not have two-digits is transferred to the output field from left to right. Blank characters are omitted.

Example

Conversion from the external into the internal format:

1. '12' → '2012'
2. '51' → '1951'
3. '1997' → '1997'
4. '991# → '991#

4.1.6.5.3.9.5 ISOLA Conversion Routine

Use

Conversion routine ISOLA converts the two-digit ISO language abbreviation INPUT into its SAP-internal OUTPUT presentation.

Functions

These are assigned using the LAISO and SPRAS fields in table T002. An INPUT that cannot be converted (because it is not defined as T002-LAISO) produces an error message and triggers the UNKNOWN_LANGUAGE exception.

Because they are compatible, single-digit entries are supported in that they are transferred to OUTPUT unchanged. They are not checked against table T002.

The difference between upper and lower case letters is irrelevant with two-digit entries however with single-digit entries, upper and lower case letters stand for different languages.

4.1.6.5.3.9.6 MATN1 Conversion Routine

Use

This conversion routine changes internal material numbers, stored in the system, into the external material numbers displayed in the interface and vice versa, according to settings in transaction OMSL.

With regard to the specific details of the conversion, read the help for the appropriate input field of the transaction.

4.1.6.5.3.9.7 NUMCV Conversion Routine

Functions

When converting from an external into an internal format this checks whether the entry in the INPUT field is wholly numerical, whether it consists of digits only, possibly with blank spaces before and/or after. If yes, the sequence of digits is copied to the OUTPUT field, right-aligned, and the space on the left is filled with zeros (‘0’). Otherwise the blank characters are removed from the sequence of digits, the result is transferred, left-aligned, into the output field, and this is then filled from the right with blank characters.

Converting from the internal format into the external format (conversion routine CONVERSION_EXIT_NUMCV_OUTPUT) does not produce any changes. The output field is set the same as the input field.

Example

Input and output fields are each 8 characters long. A conversion from an external to an internal format takes place:

1. '1234 ' → '00001234'
2. 'ABCD ' → 'ABCD '
3. ' 1234 ' → '00001234'
4. ’ AB CD’ → ’ABCD ’

4.1.6.5.3.9.8 PERI5 Conversion Routine

Use

The PERI5 conversion serves to convert a five-figure calendar quarter in an external format (Q.YYYY, for example) into the internal format (YYYYQ). Y stands for year (here four digits) and Q for quarter (single digit: 1,2,3, or 4). The separator (‘.’ or ‘/’) has to correspond to the date format in the user settings.

Features

Permitted entries for the date format DD.MM.YYYY are QYY (two digits for year without separator), Q.YY (two digits for year with separator), QYYYY (four digits for year without separator), and Q.YYYY (four digits for year with separator). Permitted entries for the date format YYYY/MM/DD would be YYQ, YY/Q, YYYYQ, YYYY/Q.

Example

Examples where the date format in the user settings is DD.MM.YYYY. A conversion from the external to the internal format takes place:

1. '2.02' → '20022'
2. '31999' → '19993'
3. '4.2001' → '20014'

4.1.6.5.3.9.9 PERI6 Conversion Routine

Use

Conversion routine PERI6 is used with six-digit entries for (fiscal year) periods.

Features

The internal format for six-digit periods is YYYYPP (200206, for example, for period 06 of fiscal year 2002). When the external format is converted to the internal format, this checks whether the entries in the INPUT parameter with external date format (separators, order) comply with user settings. The separator (‘.’ or ‘/’) has to correspond to the date format in the user settings.

Different abbreviated entries are possible and these are converted correctly into the internal format.

Example

For the external date format DD.MM.YYYY in the user settings, the following conversion takes place from external to internal formats:

1. '12.1999' → '199912'
2. '1.1999' → '199901'
3. '12.99' → '199912'
4. '1.99' → '199901'

4.1.6.5.3.9.10 PERI7 Conversion Routine

Use

Conversion routine PERI7 is used with seven-digit entries for (fiscal year) periods.

Features

The internal format for seven-digit periods is YYYYPPP (2002006, for example, for period 006 of fiscal year 2002). When the external format is converted to the internal format, this checks whether the entries in the INPUT parameter with external date format (separators, order) comply with user settings. The separator (‘.’ or ‘/’) has to correspond to the date format in the user settings.

Different abbreviated entries are possible and these are converted correctly into the internal format.

Example

For the external date format DD.MM.YYYY in the user settings, the following conversion takes place from external to internal formats:

1. '012.1999' → '1999012'
2. '12.1999' → '1999012'
3. '1.1999' → '1999001'

4. '012.99' → '1999012'
5. '12.99' → '1999012'
6. '1.99' → '1999001'

4.1.6.5.3.9.11 POSID Conversion Routine

Use

The POSID conversion routine converts the external presentation of the program position (0PROG_PO_EX) into the internal presentation (0PROG_POS), using the active master data table entries for the program position InfoObject (0PROG_POS).

Example

Conversion from the external into the internal format:

P-2411 → P24110000

4.1.6.5.3.9.12 PROJ Conversion Routine

Use

There are extensive possibilities in the ERP system project system for editing the external presentation of the project and PSP elements (project coding, editing mask). These features are included in the ERP conversion routine. This comprehensive logic cannot be mapped in the BI system. For this reason, the characteristic 0PROJECT_EX exists in the attributes of InfoObject 0PROJECT and the external description is stored there. As the external description is entered on the screen, conversion routine 'CONVERSION_EXIT_PROJ_INPUT' reads the corresponding internal description 0PROJECT and uses this for internal processing.

If no master data has been loaded into the BI system (master data generated by uploading transaction data), then the internal description has to be input in order to execute a query.

Example

Internal format: 0PROJECT: ‘A0001’

External format: 0PROJECT_EX: 'A / 0001'

4.1.6.5.3.9.13 REQID Conversion Routine

Use

The REQID conversion routine converts the external presentation of the appropriation request (0APPR_REQU) into the internal presentation (0APPR_RE_ED), using the active entries in the master data table for the appropriation request InfoObject (0APPR_RE_ED).

Example

Conversion from the external into the internal format:

P-2411-2 → P24110002

4.1.6.5.3.9.14 IDATE Conversion Routine

Use

This conversion routine assigns the appropriate internal date presentation (YYYYMMDD) to an external date (01JAN1994, for example).

Call up the test report RSSCA1T0 to be able to better visualize the functionality of this routine. This test report contains the complete date conversion with external as well as internal presentations.

Example

Conversion from the external into the internal format:

'02JAN1994' → '19940102'

4.1.6.5.3.9.15 Conversion Routine RSDAT

Use

Converts a date in an external format into the internal format.

Features

First, the system tries to convert the date in accordance with the user settings (*System* → *User Profile* → *Own Data* → *Fixed Values* → *Date Format*). If the system cannot perform the conversion in this way, it automatically tries to identify the format.

Valid formats:

DD.MM:YYYY

MM/DD/YYYY

MM-DD-YYYY

YYYY.MM.DD

YYYY/MM/DD

YYYY-MM-DD

For automatic recognition, the year has to be in four-digit format. If the date is specified as **`14.4.72`**, this is not unique and can cause errors.

Note: If the system can sensibly specify a date from the format in the user settings, this conversion is performed.

In this example, if the format in the user settings is DD.MM.YYYY, the date is converted to 19720414, since the system conversion recognizes the date.

Example

Conversion from an external format into an internal format:

4/14/1972 → 19710414

4.1.6.5.3.9.16 SDATE Conversion Routine

Use

This conversion routine assigns the appropriate internal date presentation (YYYYMMDD) to an external date (01.JAN.1994, for example).

Call up the test report RSSCA1T0 to be able to better visualize the functionality of this routine. This test report contains the complete date conversion with external as well as internal presentations.

Example

Date formatting definition in the user master record: DD.MM.YYYY

Conversion from the external into the internal format:

'02.JAN.1994' → '19940102'

4.1.6.5.3.9.17 WBSEL Conversion Routine

Use

There are extensive possibilities in the ERP system project system for editing the external presentation of the project and PSP elements (project coding, editing mask). These features are included in the ERP conversion routine. This comprehensive logic cannot be mapped in the BI system. For this reason, the characteristic 0WBS_ELM_EX exists in the attributes of InfoObject 0WBS_ELEMT and the external description is stored there. As the external description is entered on the screen, conversion routine 'CONVERSION_EXIT_WBSEL_INPUT' reads the corresponding internal description 0WBS_ELEMT and uses this for internal processing.

If no master data has been loaded into the BI system (master data generated by uploading transaction data), then the internal description has to be input in order to execute a query.

Example

Internal Format: 0WBS_ELEMT: 'A0001-1'

External format: 0WBS_ELM_EX: 'A / 0001-1'

4.1.6.5.4 Creating InfoObjects: Key Figures

Procedure

1. In the context menu of your InfoObject catalog for key figures, select *Create InfoObject*.
2. Enter a name and a description
3. If necessary, define a reference key figure or a template InfoObject.

   Template InfoObject: If you choose a template InfoObject, copy its properties to your new key figure so that you can edit them.

Reference key figure: With a reference key figure, the original value is filled from the referenced key figure. However, it is calculated differently with this key figure (either with other aggregations or with Elimination of Internal Business Volume [External] in the query). When creating update rules, a key figure with a reference is not offered. Therefore, it is not possible to create update rules.

4. Confirm your entries.
5. Edit Tab Page: Type/Units [Page 107].
6. Edit Tab Page: Aggregations [Page 108].
7. Edit Tab Page: Additional Properties [Page 110].
8. If you created your key figure with a reference, you get an additional *Elimination* tab page.
9. Save and activate the key figure you have created.

   Key figures have to be *activated* before they can be used.

   Save means that all changed key figures in the InfoObject catalog are created, and that the table entries are saved. However, they cannot be used for reporting in InfoProviders yet. The older *active* version is retained initially.

   The system only creates the corresponding data dictionary objects (data elements, domains, programs) after you have *activated* the key figure. Only then do the InfoProviders use the activated, new version.

4.1.6.5.4.1 Tab Page: Type/Unit

Functions

Key Figure Type

Specify the *type*. Amounts and quantities need unit fields.

Data Type

Specify the *data type*. For the amount, quantity, and number, you can choose between the decimal number and the floating point number, which guarantees more accuracy. For the key figures date and time, you can choose the decimal display to apply to the fields.

The following combinations of key figure and data type are possible:

Key Figure Type	Data Type
AMO Amount	CURR: Currency field, created as DEC
	FLTP: Floating point number with 8 byte precision
QUA Quantity	QUAN: Quantity field, created as DEC
	FLTP: Floating point number with 8 byte precision
NUM Number	DEC: Calculation or amount field with comma and +/- sign.
	FLTP: Floating point number with 8 byte precision
INT integer	INT4: 4 byte integer, whole number with +/- sign
DAT Date	DATS: Date field (YYYYMMDD), created as char(8)
	DEC: Calculation or amount field with comma and +/- sign.
TIM Time	TIMS: Time field (hhmmss), created as char(8)
	DEC: Calculation or amount field with comma and +/- sign.

Currency/Quantity Unit

You can assign a *fixed currency* to the key figure. If this field is filled, the key figure bears this currency throughout BW.

You can also assign a variable currency to the key figure. In the field *unit/currency*, determine which InfoObject bears the key figure unit. For quantities or amount key figures, either this field must be filled, or you must enter a fixed currency or amount unit.

4.1.6.5.4.2 Tab Page: Aggregation

Functions

Aggregation:

There are three aggregation options:

- Minimum (MIN): The minimum value of all values displayed in this column is displayed in the results row.
- Maximum (MAX): The maximum value of all values displayed in this column is displayed in the results row.
- Summation (SUM): The sum of all values displayed in this column is displayed in the results row.

Exception aggregation

This field determines how the key figure is aggregated in the Business Explorer in relation to the exception characteristic. This reference characteristic must be unique in the query. In general, this refers to time.

The key figure *Number of Employees* would, for example, be totaled using the characteristic *Cost Center*, and not a time characteristic. Here you would determine a time characteristic as an exception characteristic with, for example, the aggregation *Last Value*...

You can find more extensive examples in Scenarios for Using Exception Aggregation [External].

The following exception aggregations are possible:

- Average (value not equal to zero) (AV0): After drilling down according to the reference characteristic, the average of the column value not equal to zero is displayed in the results row.
- Average (weighted with no. of days) (AV1): After drilling down according to the reference characteristic, the average of the column value weighted with the number of days is displayed in the results row.
- Average (weighted with the number of workdays; factory calendar) (AV2): After drilling down according to the reference characteristic, the average of the column value weighted with the number of workdays is displayed in the results row.
- Average (all values) (AVG): The average of all values is displayed.
- Counter (value not equal to zero) (CN0): The number of values <> zero is displayed in the results row.
- Counter (all values) (CNT): The number of existing values is displayed in the results row.
- First value (FIR): The first value in relation to the reference characteristic is displayed in the results row.
- Last value (LAS): The last value in relation to the reference characteristic is displayed in the results row.
- Maximum (MAX): see above
- Minimum (MIN): see above
- No aggregation (exception if more than one record arises)
- No aggregation (exception if more than one value arises)
- No aggregation (exception if more than one value <> 0)
- Standard deviation (STD): After drilling down according to the reference characteristic, the standard deviation of the displayed values is displayed in the results row.
- Summation (SUM): see above
- Variance (VAR): After drilling down according to the reference characteristic, the variance of the displayed values is displayed in the results row.

See also Aggregation Behavior of Non-Cumulative Key Figures [Page 423].

Referenced characteristic for exception aggregation

In this field, select the characteristic in relation to which the key figure is to be aggregated with the exception aggregation. Often this is a time characteristic. However, you can use any characteristic you wish.

Flow/non-cumulative value

You can select the key figure as a *cumulative value*. Values for this key figure have to be posted in each time unit, for which values for this key figure are to be reported.

Non-cumulative with non-cumulative change

The key figure is a non-cumulative. You have to enter a key figure that represents the non-cumulative change of the non-cumulative value. There do not have to be values for this key figure in every time unit. For the non-cumulative key figure, values are only stored for selected times (markers). The values for the remaining times are calculated from the value in a marker and the intermediary non-cumulative changes.

Non-Cumulative with inflow and outflow

The key figure is a non-cumulative. You have to specify two key figures that represent the inflow and outflow of a non-cumulative value.

For non-cumulatives with non-cumulative change, or inflow and outflow, the two key figures themselves are not allowed to be non-cumulative values, but must represent cumulative values. They must be the same type (for example, amount, and quantity) as the non-cumulative value.

See also:

Aggregation [External]

Modeling Non-Cumulatives with Non-Cumulative Key Figures [Page 420]

4.1.6.5.4.3 Tab Page: Additional Properties

Functions

Business Explorer

You can make the following settings in part specific to data target for the InfoObjects contained in the data target. The settings are then only valid in the respective data target. See also Additional Functions in InfoCube Maintenance [Page 133] and Additional Functions of ODS Object Maintenance [Page 158].

Decimal places

You can determine how many decimal places the field has by default in the Business Explorer. This layout can be overwritten in queries.

Layout

This field describes with which scaling the field in the Business Explorer is displayed by default. This layout can be overwritten in queries.

For more information, see Hierarchy Intervals [External]

Miscellaneous:

Key figure with maximum accuracy

If you choose this indicator, then the OLAP processor calculates internally with packed numbers that have 31 decimal places. In doing so, a greater accuracy is attained, and the rounding differences are reduced. Normally, the OLAP processor calculates with floating point numbers.

Exclusive attributes:

If you choose *exclusively attribute*, then the created key figure can only be used as an attribute for another characteristic, but cannot be used as a dedicated key figure in the InfoCube.

4.1.6.5.5 Editing InfoObjects

Prerequisites

You have already created an InfoObject.

See also:

Creating InfoObjects: Characteristics [Page 60]

Creating InfoObjects: Key Figures [Page 107]

Procedure

You are in the Data Warehousing Workbench in the modeling view of the InfoObject tree.

Select the InfoObject you want to maintain, and, using the context menu, choose *Change*. Alternatively, select the InfoObject you want to maintain, and choose the *Maintain InfoObjects* icon from the menu toolbar. You get to the InfoObject maintenance.

Change Options

It is usually possible to change the meaning and the text of an InfoObject. However, only limited changes can be made to certain properties if the InfoObject is used in InfoProviders.

With key figures, for example, you cannot change the key figure type, the data type, or the aggregation, as long as the key figure is still being used in an InfoProvider. Use the *Check* function to get hints on incompatible changes.

With characteristics, you can change compounding and data type, but only if no master data exists yet.

You cannot delete characteristics that are still in use in an InfoProvider, an InfoSource, compounding or as an attribute. It is a good idea, therefore, to execute a where-used list, whenever you want to delete a characteristic. If the characteristic is being used, you have to first delete the InfoProvider or the InfoObject from the InfoProvider. If errors occur, or applications exist, an error log appears automatically.

4.1.6.5.6 Additional Functions in InfoObject Maintenance

Functions

There are other functions available in the InfoObject maintenance in addition to creating, changing, and deleting InfoObjects.

Documents

This function allows you to display, create or change documents for InfoObjects.

See: **Documents [External]**.

Display in Tree

Use this function to display, in a clear tree structure, all the settings for an InfoObject that have been made in the InfoObject maintenance tab pages.

Version Comparison

This function compares the following InfoObject versions:

- the active and revised versions of an InfoObject
- the active and Content versions of an InfoObject
- the revised and Content versions of an InfoObject

This enables you to compare all the settings made in the InfoObject maintenance tab pages.

Transport Connection

You can choose and transport InfoObjects. All BW Objects that are needed to ensure a consistent status in the target system are collected automatically.

Where-Used List

You determine which other objects in BW also use a specific InfoObject.

You can determine what effects changing an InfoObject in a particular way will have, and whether this change is permitted at the moment or not.

Analyzing InfoObjects

You get to the analysis and repair environment by choosing *Edit InfoObject* from the main menu. You can use the analysis and repair environment to check the consistency of your InfoObjects.

See Analysis and Repair Environment [Page 792]

Object Browser Using AWB

Using this function in the main menu by means of *Environment* → *Object Browser Using AWB*, you can display the connection between the different BW objects.

For example:

- Structural dependencies, for example the InfoObjects, from which an InfoCube is structured.

- Connections between BW objects, such as the data flow from a source system across an InfoCube to a query.

The dependencies can be displayed and exported in HTML format.

Hyperlinks

Technical objects, such as data elements or attributes, are often underlined in the InfoObject maintenance. In this case, use the context menu (right mouse click) to call up a selection of possible functions, including jumping to the detail view (dictionary), table contents, table type, and so on. Double-click to get to the detail display.

Activating in the Background

In some cases (for example when converting large amounts of data), activating an InfoObject can take a very long time. The activation process terminates after a specified amount of time. In these cases, you can activate InfoObjects with the help of a background job. In the InfoObject maintenance, choose *Characteristic* → *Activate in Background*.

Maintaining Database Save Parameters

With characteristics: Use this setting to determine how the system handles the table when it creates it in the database: You can access the function using *Extras* in the main menu. For more information, see DB Save Parameters [Page 659]

4.1.6.6 se of Master Data and Master Data-Bearing Characteristics

Definition

Master data is data that remains unchanged over a long period of time. Master data contains information that is needed again and again in the same way. Characteristics [Page 57] can bear master data in BI. Master data can be attributes, texts, or hierarchies.

If characteristics have attributes, texts, or hierarchies, they are referred to as **master data-bearing characteristics**.

The master data of a cost center contains the name, the person responsible, the relevant hierarchy area, and so on.

The master data of a supplier contains the supplier's name, address, and bank details.

The master data of a user in the SAP system contains his/her access authorizations to the system, standard printer, start transaction, and so on.

Use

When you create a characteristic InfoObject, it is possible to assign attributes, texts, hierarchies, or a combination of this master data to the characteristic. If a characteristic bears master data, you can edit it in the BI system in master data maintenance.

More information: Creating InfoObjects: Characteristics [Page 60]

You can flag a characteristic as an InfoProvider if it has attributes and/or texts. The characteristic is then available as an InfoProvider for analysis and reporting purposes.

More information: InfoObjects As InfoProvider [Page 165]

4.1.6.6.1 Master Data Types: Attributes, Texts and Hierarchies

Use

There are three different types of master data in BI:

Attributes

Attributes are InfoObjects that are logically subordinate to a characteristic. You cannot select attributes in the query.

You assign the attributes *Person responsible for the cost center* and *Telephone number of the person responsible for the cost center* (characteristics as attributes), as well as *Size of the cost center in square meters* (key figure as attribute) to a *Cost Center*.

Texts

You can create text descriptions for master data or load text descriptions for master data into BI. Texts are stored in a text table.

In the text table, the *Name of the person responsible for the cost center* is assigned to the master data *Person responsible for the cost center*.

Hierarchies

A hierarchy serves as a context and structure for a characteristic according to individual sort criteria. For more detailed information, see Hierarchies [External].

Features

Time-dependent attributes:

If the characteristic has at least one time-dependent attribute, a time interval is specified for this attribute. Since the time frame for master data on the database must always be between 01.01.1000 and 12.31.1000, the gaps are filled automatically (see Maintaining Time-Dependent Master Data [Page 116]).

Time-dependent texts:

If you create time-dependent texts, the system always displays the text for the key date in the query.

Time-dependent texts and attributes:

If texts and attributes are time dependent, the time intervals do not have to agree.

Language-dependent texts:

In Characteristic InfoObject Maintenance [Page 60], you specify whether texts are language specific (for example, with product names: German → Auto, English → car) or are not language specific (for example, customer names). The system only displays texts in the selected language.

If texts are language dependent, you have to load all texts with a language indicator.

Only texts exist:

You can also create texts only for a characteristic, without maintaining attributes. When you load texts, the system automatically generates the entries in the SID table.

4.1.6.6.2 Master Data Maintenance

Use

Master data maintenance allows you to change or regenerate master data attributes or texts manually in BW. Data is always maintained per characteristic.

There are two different master data maintenance sessions:

- Creating or Changing Master Data [Page 115]
- Deleting Master Data at Single Record Level [Page 119]

Integration

You cannot run the two sessions at the same time. This means that

- if you choose the *Change* function in the master data maintenance screen, the deletion function is deactivated, and is only reactivated once you have saved your changes.
- if you select a master data record in the master data maintenance screen and choose the *Delete* function, the create and change functions are deactivated, and are only reactivated once you have finished deleting the record and clicked *Save*.

Functions

Creating or Changing Master Data:

You can add new master data records to a characteristic, change individual master data records, or select several master data records and assign global changes to them.

Deleting Master Data at Single Record Level:

You can delete individual records or select and delete several records.

You can only delete master data records if no transaction data exists for the master data that you want to delete, the master data is not used as attributes for an InfoObject, and there are no hierarchies for this master data.

4.1.6.6.2.1 Creating and Changing Master Data

Prerequisites

If master data is maintained for a master data-bearing characteristic, you can modify this master data and create additional master data records.

Procedure

1. Navigate to master data maintenance by choosing

   InfoObject Tree → InfoObject → Context Menu (secondary mouse button) → Maintain Master Data.

   or from InfoObject maintenance by choosing *Maintain Master Data*

   A selection screen appears for restricting the master data you want to edit.

2. Using the options from the F4 Help, select the relevant data.

   The list overview of the selection appears. The list overview is also displayed if no hits were found for your selection, so that you can enter new master records for particular criteria.

3. Make your changes with the help of the relevant maintenance function.

   Creating new master records

   Choose *Create* to add new master records. New records are added to the end of the list.

   Changing single records

   Double-clicking a data record takes you to the individual maintenance. Make the relevant changes in the subsequent change dialog box.

   Mass changes

   Select multiple records, and choose *Change.* A change dialog box appears in which the attributes and texts are offered. Enter the relevant entries, which are then transferred to all the selected records.

4. Choose *Save*.

   If a newly created record already exists in the database but does not appear in the processing list (because you have not selected it on the selection screen), there is no check. Instead, the old records are overwritten.

   If you change master data in the BI system, you must adjust the respective source system accordingly. Otherwise the changes will be overwritten in the BI system the next time data is uploaded.

   Master data that you have created in the BI system is retained even after you have uploaded data from the source system.

Note the exception for time-dependent master data [Page 116].

4.1.6.6.2.1.1 Maintaining Time-Dependent Master Data

Use

The maintenance of the master data is more complex with time-dependent master data, as the validity period of a text is not necessarily in concordance with that of an attribute master record.

The InfoObject **User master record** has the time-dependent attribute **Personnel number**, and the time-dependent text **User name**. If the user name

changes (after marriage, for example), the personal number still remains the same.

Prerequisites

In the InfoObject Maintenance [Page 60], make sure that the relevant InfoObject is flagged as ‘time-dependent’.

Procedure

To maintain texts with time-dependent master data, proceed as follows:

1. Select the master data that want, and select one of the three text pushbuttons.

   If you choose ***Display text***, a list appears containing all the texts for this characteristic value. By double clicking, you can select a text. A dialog box appears with the selected text for the characteristic value.

   If you choose ***Change text***, a list appears containing all the texts for this characteristic value. By double clicking, you can select a text. A dialog box appears with the selected text for the characteristic value, which you can then edit.

   If you choose ***Create text***, a dialog box appears in which you can enter a new text for the characteristic value.

   The texts always refer to the selected characteristic value.

2. Choose *Save.*

   When you select time-dependent master data with attributes, the list displays the texts that are valid until the end of the validity period of the characteristic value. When you change and enter new texts, the lists are updated.

   Master data must exist between the period of 01.01.1000 and 12.31.1000 in the database. When you create data, gaps are automatically filled. When you change or initially create master data, in some cases, you must adjust the validity periods of the adjoining records accordingly.

   If a newly created record already exists in the database but does not appear in the processing list (because you have not selected it in the selection screen) there is no check. Instead, the old records are overwritten.

   If you change master data in BW, you must adjust the respective source system accordingly. Otherwise the changes will be overwritten in BW the next time you upload data.

   Master data, which you have created in BW, remains even after you have uploaded data from the source system.

4.1.6.6.2.1.2 Time-Dependent Master Data from Different Systems

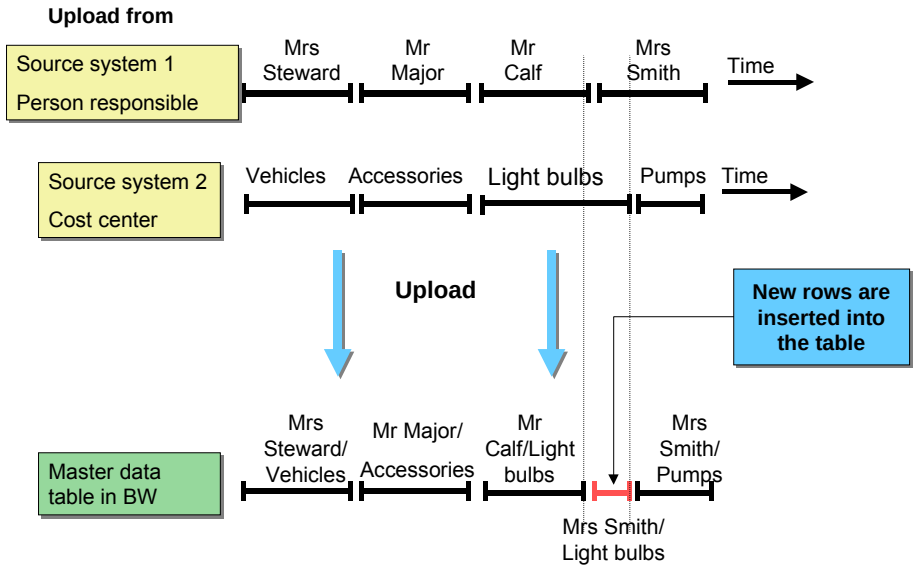
Use

You have the option of uploading time-dependent characteristic attributes from different systems, even if the time intervals of the attributes are different.

Functions

If you load time-dependent characteristic attributes from different source systems, these are written in the master data table, even if the time intervals are different.

From source system 1, load attribute A with the values 10, 20, 30 and 40. From source system B, load attribute B with the values 15, 25, 35 and 45. The time intervals of the last two values are different.



The system inserts another row into the master data table:

dateto	datefrom	Person Responsible	Cost Center
01.01.1999	28.02.1999	Mrs Steward	Vehicles
01.03.1999	31.05.1999	Mr Major	Accessories
01.06.1999	31.08.1999	Mr Calf	Light bulbs
01.09.1999	10.09.1999	Mrs Smith	Light bulbs
11.09.1999	30.09.1999	Mrs Smith	Pumps

4.1.6.6.2.2 Deleting Master Data at Single Record Level

Use

Besides creating and changing master data, you can also use a deletion mode at single record level.

Procedure

1. You navigate to deletion mode from master data maintenance by choosing

   InfoObject Tree → InfoObject → Context Menu (secondary mouse button) → Maintain Master Data.

   or from InfoObject maintenance by choosing *Maintain Master Data.*

   A selection screen appears for restricting the master data you want to edit.
2. Using the options from the input help, select the relevant data.
3. The list overview for the selection is displayed, and provides two options:
   - In the list, select the master data records to be deleted, choose *Delete,* and *Save* your entries.
   - Select additional master data by choosing *Data Selection,* select the master data records that are to be deleted, and choose *Delete.* Repeat the selection as necessary and choose *Save* to finish.

   The records marked for deletion are first written into the deletion buffer. If you choose *Save,* the system generates a where-used list for the records marked for deletion. Master data that is not being used in other objects is deleted.

4.1.6.6.3 Deleting Master Data and Texts for a Characteristic

Use

You can delete master data and texts directly from the master data table in BW. In contrast to deleting at single record level, you can use this function to delete all the existing master data and texts for a characteristic in one action.

Prerequisites

In order to delete master data there must be no transaction data in BW for the master data in question, it must not be used as an attribute for InfoObjects and there must not be any hierarchies for this master data.

Functions

You reach the *Delete Master Data* function from the context menu of your InfoObject in the InfoObject tree and also the InfoSource tree.

If you choose the *Delete Master Data* function, the program checks the entries in the master data table affected to see if they are used in other objects.

When you delete you are able to choose whether entries in the SID table of a characteristic are to be retained or whether they are to be deleted:

If you delete the SID table entry for a particular characteristic value, the SID value assigned to the characteristic value is lost. If you load new attributes for this characteristic value later, a new SID value has to be created for the characteristic value. In general this has a negative effect on the runtime required for loading. In some cases deleting entries from the SID table can also lead to serious data inconsistencies. This occurs if the list of SID values generated from the where-used list is not comprehensive, however, this is rare.

Delete, retaining SIDs

For the reasons given above, you should choose this option as standard. Even if, for example, you want to make sure that individual attributes of the characteristic that are no longer needed are deleted before you load master data attributes or texts, the option of deleting master data but retaining the entries from the SID table is also absolutely adequate.

Delete with SIDs

Note that deleting entries from the SID table is only necessary, or useful, in exceptional cases. Deleting entries from the SID table does make sense if, for example, the composition of the characteristic key is fundamentally changed and you want to swap a large record of characteristic values with a new record with new key values.

4.1.6.6.4 Versioning Master Data

Attributes and hierarchies are available in two versions, an active (A) version and a modified (M) version. Texts are active immediately after they have been loaded. Existing texts are overwritten when new texts are loaded.

Attribute versions are managed in the P table [External] and in the Q table [External]. Time-independent attributes are stored in the P table and time-dependent attributes are stored in the Q table. From left to right, the P table contains the key fields of the characteristic (for example, 0COSTCENTER: CO_AREA and COSTCENTER), the technical key field OBJVERS (versioning), the indicator field CHANGED ID (versioning), and 0 or more attribute fields that can be display attributes or navigation attributes. The structure of the Q table is identical to the structure of the P table, with the addition of the 0DATEFROM and 0DATETO fields to map the time-dependency.

The OBJVERS and CHANGED fields must always be taken into account in versioning:

If you load master data that does not yet exist, an active version of this data is added to the table. If, when you reload the data, the value of the attribute changes, the active entry is flagged *for deletion* (CHANGED = D) and the M/I (modified(insert)) version of the new record is added.

You are loading master data for the 0COSTCENTER characteristic. After activation, the P table looks like this:

CO_AREA	COSTCENTER	OBJVERS	CHANGED	RESP_PERS
1000	0000001000	A		Patel
1000	0000001020	A		Jones

Later, you load new records. These new records are given the OBJVERS entry M and the CHANGED entry I. The available records, for which new data has been loaded, are given the OBJVERS entry D for “to be deleted”:

CO_AREA	COSTCENTER	OBJVERS	CHANGED	RESP_PERS
1000	0000001000	A		Patel
1000	0000001020	A	D	Jones
1000	0000001020	M	I	Smith

Before the new records can be displayed in reporting, you have to start the change run (see System Response Upon Changes to Data: Aggregate [Page 689]). During the change run, the old record is deleted and the new record is set to active.

BI reporting always reads the active version. InfoSets are an exception to this rule, as the *most recent reporting* function can be switched on in the InfoSet Builder. In such an InfoSet, the most recent records are displayed in reporting, even if they are not yet active.

For more information, see Most Recent Reporting for InfoObjects [Page 205].

4.1.6.6.5 Activating Master Data and Texts

Prerequisites

Master data and texts have already been loaded into the BI system using the scheduler [Page 342].

Procedure

Activating Master Data

When you update master data from an SAP system, the master data is imported in an inactive state. You must activate the new master data so that it can be accessed and used for reporting purposes.

More information: Versioning Master Data [Page 120]

Choose *InfoObject Tree* → *Context Menu of Corresponding Characteristic* → *Activate Master Data.*

Upon activation, there are two scenarios to choose from:

The master data is already being used in aggregates in the InfoCube:

If you are already using the existing master data in aggregates in InfoCubes, you cannot activate the master data individually. In this case, proceed as follows:

1. In the main menu, choose *Tools* → *Hierarchy/Attribute Changes.*
2. Execute the change run.

   More information: System Response Upon Changes to Master Data and Hierarchies [Page 689]

   The system now automatically restructures and activates the master data and its aggregates.

Note that this process can take several hours if there is a high volume of data. You should therefore simultaneously activate all the characteristics that are affected by changes to their master data at regular intervals.

The corresponding master data is not being used in aggregates:

Choose *InfoObject Tree* → *Context Menu of Corresponding Characteristic* → *Activate.*

The system now automatically activates the master data so that it can be used directly in reporting.

Activating Texts

Texts are active immediately and can be used directly in analysis and reporting. You do not need to activate them manually.

4.1.6.6.6 Simulate Loading of Master Data

Use

This function allows you to simulate the loading of a master data packet before loading the data into BW. This means you can be aware of errors in the data loading early on and remove problems in advance.

Integration

You can get to the function from the *Monitor Detailed Screen* → Tabstrip *Updated Data* or *All IDocs.*

Features

In the case of data without errors, the loading simulation provides you with a detailed description of the processes that are run during loading. The left-hand frame structures the various master data types that can be loaded in a tree. Either:

- Time-dependent texts and/or time-constant texts, or
- Time-dependent master data attributes and/or time-constant texts

On the level below the master data types, you will see the different database operations that are carried out during loading. (For example, modifying, inserting, deleting)

By clicking on a master data type or on a database operation, or by using drag-and-drop on these objects in the right-hand frame, you can obtain a detailed view of the respective uploaded data.

In the case of incorrect data, only the master data types, and not the database operations, are displayed in the left-hand frame.

The corresponding error log appears in the lower frame.

4.1.6.6.7 Master Data Lock

Use

During the master data load procedure, the master data tables concerned are locked so that, for example, data cannot be loaded at the same time from different source systems, which would bring about inconsistencies.

In certain cases, for example if a program termination occurs during the load process, then the locks are not automatically removed after the load process.

You then have to manually delete the master data locks.

Activities

You get to the master data lock overview via the padlock symbol. Using the context menu (right mouse button), choose the corresponding master data symbol, and delete the master data lock.

4.1.6.6.8 Reorganizing Master Data

Use

You can reorganize the dataset for texts and attributes belonging to a basic characteristic. The reorganization process finds and removes redundant data records in the attribute- and text- tables. This reduces the volume of data and improves performance.

Functions

For a given basic characteristic, the system firstly compares data in the active and modified versions of the time-dependent and non-time-dependent attributes with each other. If there are no differences between the active and the modified versions, the redundant data is compressed. In a second step, the system checks time-dependent texts and attributes to see whether time intervals exist with identical attribute values or text entries. If this is the case, the affected time intervals are combined into larger intervals.

Firstly, the attribute *Cost Center Manager* (0RES_PERSON) is changed, as the only attribute, for a *Cost Center* and then is reset to its original value using a second load process. Therefore, the name of the Cost Center Manager has not actually changed. In this case, the reorganization means that the data record is deleted for the changed version (M version).

For a *Cost Center*, the same person is entered as *Cost Center Manager* for the period 01.06.2001-31.12.2001 as for the period 01.01.2002 – 31.03.2002. The process of reorganization combines these two intervals into one, providing that the other time-dependent attributes for the cost center are consistent across both intervals.

0COSTCENTER

dateto	datefrom	bus_area	comp_code	res_pers
31.12.2001	01.06.2001	1400	1000	Jones
31.03.2002	01.01.2002	1400	1000	Jones

You can carry out the master data reorganization process as a process type in process chain maintenance.

Activities

During master data organization for attributes and texts, the system sets locks preventing access to the basic characteristic currently being processed. These locks correspond to the locks preventing the loading of the master data attributes and texts. This means that it is not possible to load, delete or change master data for this characteristic during the reorganization process.

When assigning locks, the system distinguishes between locks for attributes and locks for texts. This means that you can load texts for this characteristic during a reorganization that only affects attributes, and vice versa.

4.1.6.6.9 oad Master Data to InfoProviders Straight from Source Systems

In data transfer process (DTP) maintenance, you can specify that data is not extracted from the PSA of the DataSource but is requested straight from the data source at DTP runtime. The *Do not extract from PSA but allow direct access to data source* indicator is displayed for the *Full* extraction mode if the source of the DTP is a DataSource. We recommend that you only use this indicator for small datasets; small sets of master data, in particular.

Extraction is based on synchronous direct access to the DataSource. The data is not displayed in a query, as is usual with direct access, but is updated straight to a data target without being saved in the PSA.

Dependencies

If you set this indicator, you do not require an InfoPackage to extract data from the source.

Note that if you are extracting data from a file source system, the data is available on the application server.

Using the *Direct Access* mode for extraction has the following implications, especially for SAP source systems (SAPI extraction):

- Data is extracted synchronously. This places a particular demand on the main memory, especially in the source system.
- The SAPI extractors may respond differently than during asynchronous load since they receive information by direct access.

- SAPI customer enhancements are not processed. Fields that have been added using the append technology of the DataSource remain empty. The exits RSAP0001, exit_saplrsap_001, exit_saplrsap_002, exit_saplrsap_004 do not run.
- If errors occur during processing in BI, you have to extract the data again since the PSA is not available as a buffer. This means that deltas are not possible.
- In the DTP, the filter only contains fields that the DataSource allows as selection fields. With an intermediary PSA, you can filter by any field in the DTP.

4.1.6.7 InfoProviders

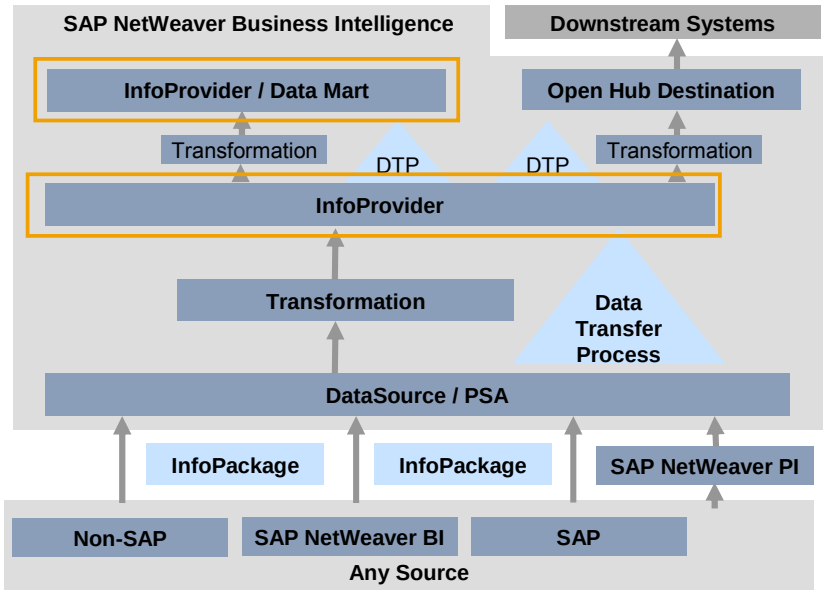
Definition

Generic term for BI objects into which data is loaded or that display views of data. You analyze this data in BEx queries.

Use

InfoProviders are different metaobjects in the data basis that can be seen within query definition as uniform data providers. Their data can be analyzed in a uniform way. The type of data staging and the degree of detail or "proximity" to the source system in the data flow diagram differs from InfoProvider to InfoProvider. However, in the BEx Query Designer, they are seen as uniform objects.

The following graphic shows how InfoProviders are integrated in the dataflow:



Structure

The term InfoProvider encompasses objects that physically contain data:

InfoCubes [Page 126]

DataStore objects [Page 148]

InfoObjects as InfoProviders [Page 165]

Staging is used to load data into these InfoProviders.

InfoProviders can also be objects that do not physically store data but which display logical views of data, such as:

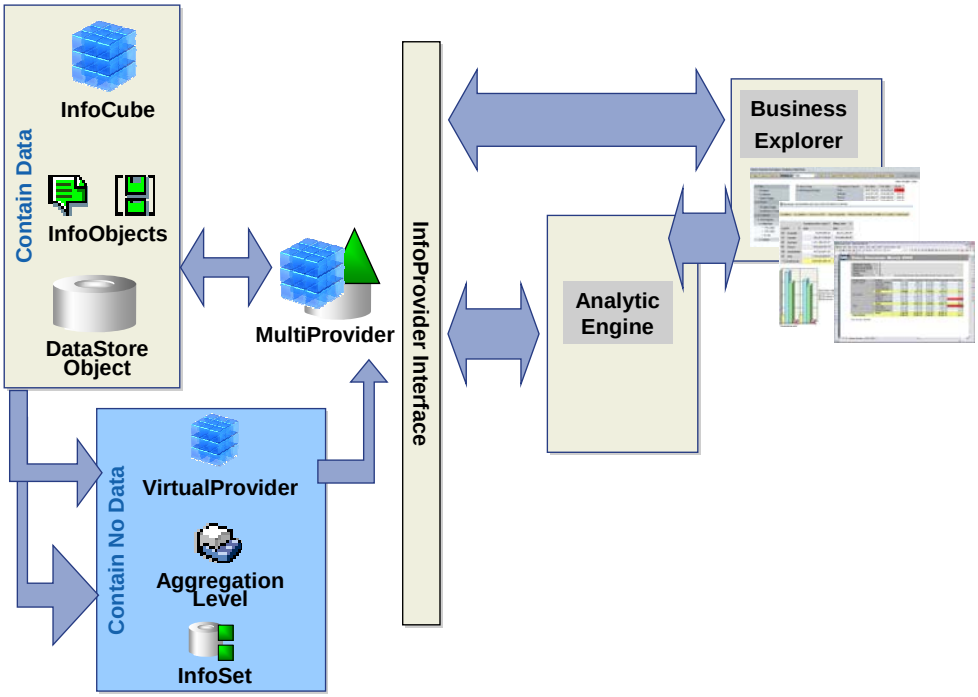
VirtualProviders [Page 165]

InfoSets [Page 176]

MultiProviders [Page 215]

Aggregation levels [External]

The following figure gives an overview of the BI objects that can be used in analysis and reporting. They are divided into InfoProviders that contain data and InfoProviders that only display logical views and do not contain any data. In BEx, the system accesses an InfoProvider; it is not important how the data is modeled.



4.1.6.7.1 nfoCubes

Definition

Type of InfoProvider.

An InfoCube describes (from an analysis point of view) a self-contained dataset, for example, for a business-orientated area. You analyze this dataset in a BEx query.

An InfoCube is a set of relational tables arranged according to the star schema: A large fact table in the middle surrounded by several dimension tables.

Use

InfoCubes are filled with data from one or more InfoSources or other InfoProviders. They are available as InfoProviders for analysis and reporting purposes.

Structure

The data is stored physically in an InfoCube. It consists of a number of InfoObjects that are filled with data from staging. It has the structure of a star schema. For more information, see Star Schema [Page 127].

The real-time characteristic can be assigned to an InfoCube. Real-time InfoCubes are used differently to standard InfoCubes. For more information, see Real-Time InfoCubes [Page 131].

Integration

In query definition [External] in the BEx Query Designer, you access the characteristics and key figures that are defined for an InfoCube.

4.1.6.7.1.1 Star Schema

Structure

InfoCubes are made up of a number of InfoObjects. All InfoObjects (characteristics and key figures) are available independent of the InfoCube. Characteristics refer to master data with their attributes and text descriptions.

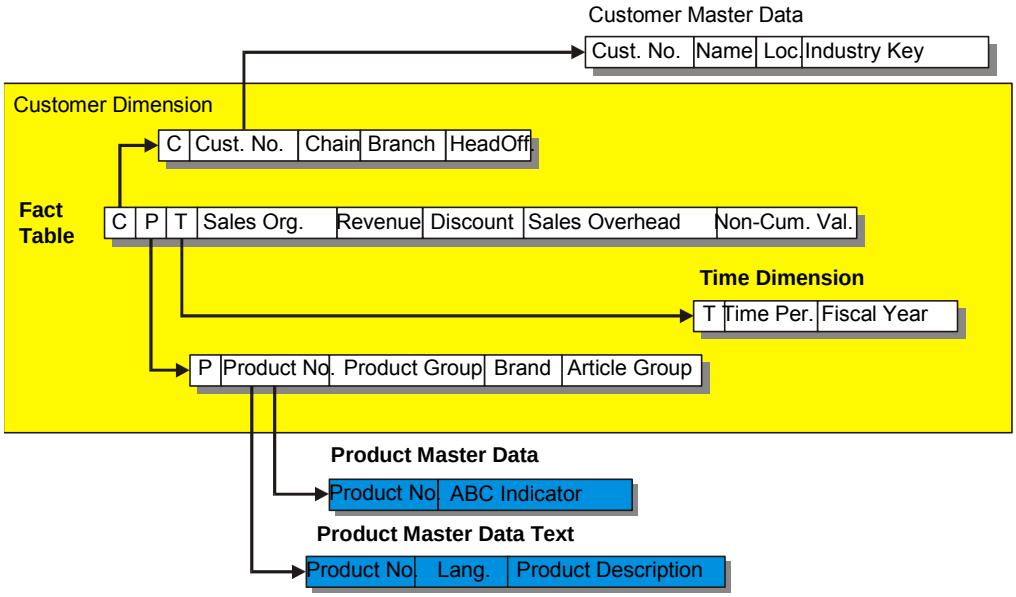
An InfoCube consists of several InfoObjects and is structured according to the star schema. This means there is a (large) fact table that contains the key figures for the InfoCube, as well as several (smaller) dimension tables which surround it. The characteristics of the InfoCube are stored in these dimensions.

An InfoCube fact table only contains key figures, in contrast to a DataStore object, whose data part can also contain characteristics. The characteristics of an InfoCube are stored in its dimensions.

The dimensions [Page 128] and the fact table are linked to one another using abstract identification numbers (dimension IDs) which are contained in the key part of the particular database table. As a result, the key figures of the InfoCube relate to the characteristics of the dimension. The characteristics determine the granularity (the degree of detail) at which the key figures are stored in the InfoCube.

Characteristics that logically belong together (for example, district and area belong to the regional dimension) are grouped together in a dimension. By adhering to this design criterion, dimensions are to a large extent independent of each other, and dimension tables remain small with regards to data volume. This is beneficial in terms of performance. This InfoCube structure is optimized for data analysis.

The fact table and dimension tables are both relational database tables.



Characteristics refer to the master data with their attributes and text descriptions. All InfoObjects (characteristics with their master data as well as key figures) are available for all InfoCubes, unlike dimensions, which represent the specific organizational form of characteristics in one InfoCube.

Integration

You can create aggregates [Page 672] to access data quickly. Here, the InfoCube data is stored redundantly and in an aggregated form.

You can either use an InfoCube directly as an InfoProvider for analysis and reporting, or use it with other InfoProviders as the basis of a MultiProvider or InfoSet.

See also:

Checking the Data Loaded in the InfoCube [Page 136]

4.1.6.7.1.2 Dimension

Definition

The grouping of classification groups (characteristics) whose content is logically connected, under a single generic term. If the dimension contains a characteristic whose value already uniquely determines the values of all other characteristics from a business-orientated viewpoint, then the dimension is named after this characteristic.

The customer dimension could, for example, be made up of the customer number, the customer group and the levels of the customer hierarchy.

The sales dimension could contain the characteristics ‘sales person’, ‘sales group’ and ‘sales office’.

The time dimension could be given using the characteristics ‘day’ (in the form YYYYMMDD), ‘week’ (in the form YYYY.WW), ‘month’ (in the form YYYY.MM), ‘year’ (in the form YYYY) and ‘period’ (in the form YYYY.PPP).

Use

When defining an InfoCube, characteristics for dimensions are grouped together to enable them to be stored in a star schema table (dimension table). The aforementioned business-orientated grouping can be the basis for this. With the aid of a simple foreign key dependency, dimensions are linked to one of the key fields of the fact table.

Also refer to Star Schema [Page 127].

When you create an InfoCube, the dimensions data package, time and unit are already defined by default. The data package dimension contains technical characteristics. Time characteristics and units are automatically assigned to the corresponding dimensions. When you activate the InfoCube, only those dimensions that contain InfoObjects are activated.

Structure

From a technical viewpoint several characteristic values are mapped to an abstract dimension key (DIM ID), to which the values in the fact table refer. The characteristics chosen for an InfoCube are divided up among InfoCube-specific dimensions when creating the InfoCube.

Also refer to the following for specific cases arising when defining dimensions:

Line Item and High Cardinality [Page 129]

4.1.6.7.1.2.1 Line Item and High Cardinality

Use

When compared to a fact table, dimensions ideally have a small cardinality. However, there is an exception to this rule. For example, there are InfoCubes in which a characteristic *Document* is used, in which case almost every entry in the fact table is assigned to a different *Document.* This means that the dimension (or the associated dimension table) has almost as many entries as the fact table itself. We refer here to a *degenerated dimension.*

Generally, relational and multi-dimensional database systems have problems to efficiently process such dimensions. You can use the indicators *line item* and *high cardinality* to execute the following optimizations:

1. *Line item*: This means the dimension contains precisely one characteristic. This means that the system does not create a dimension table. Instead, the SID table of the characteristic takes on the role of dimension table. Removing the dimension table has the following advantages:
   - When loading transaction data, no IDs are generated for the entries in the dimension table. This number range operation can compromise performance precisely in the case where a degenerated dimension is involved.
   - A table- having a very large cardinality- is removed from the star schema. As a result, the SQL-based queries are simpler. In many cases, the database optimizer can choose better execution plans.

   Nevertheless, it also has a disadvantage: A dimension marked as a *line item* cannot subsequently include additional characteristics. This is only possible with normal dimensions.

We recommend that you use DataStore objects, where possible, instead of InfoCubes for line items. See Creating DataStore Objects [Page 155].

2. *High cardinality:* This means that the dimension is to have a large number of instances (that is, a high cardinality). This information is used to carry out optimizations on a physical level in depending on the database platform. Different index types are used than is normally the case. A general rule is that a dimension has a high cardinality when the number of dimension entries is at least 20% of the fact table entries. If you are unsure, do not select a dimension having high cardinality.

Activities

When creating dimensions in the InfoCube maintenance, flag the relevant dimension as a *Line Item*/ having *High Cardinality.*

4.1.6.7.1.3 Creating InfoCubes

Prerequisites

Ensure that all the InfoObjects you want to transfer in the InfoCube are available in an active version. Create any InfoObjects you require that do not already exist and activate them.

Instead of creating a new InfoCube, you can install an InfoCube from SAP BI Content [Page 435].

Procedure

1. Create an InfoArea to which the new InfoCube should be assigned.

   To do this, choose *Modeling* → *InfoProvider*.

2. In the context menu of the InfoArea, choose *Create InfoCube*.

3. Select either *Standard* or *Real Time* as the InfoCube type. For more information, see Real-Time-Enabled InfoCube [Page 131].

   Choose *Create*.

   If you want to create a copy of an already existing InfoCube, you can enter an InfoCube as a template.

   The *Edit InfoCube* screen appears.

4. Transferring InfoObjects:

   On the left side of the screen, there are various templates to choose from. These allow you to get a better overview in relation to a particular task. For performance reasons, the default setting is an empty template. You use the pushbuttons to select different objects as templates.

   The InfoObjects that are to be added to the InfoCube are divided into the categories characteristic, time characteristic, key figure and unit. You have to transfer at least one InfoObject from each category.

   On the right side of the screen, you define the InfoCube. Use drag and drop to assign the InfoObjects in the dimensions and the *Key Figures* folder. You can select several InfoObjects at once. You can also transfer entire dimensions using drag and drop. The system assigns navigation attributes automatically. These navigation attributes can be switched on to analyze data in BEx.

Or:

You can insert InfoObjects without selecting a template in the left half of the screen. This is useful if you know exactly which InfoObjects you want to include in the InfoCube. Choose *Insert InfoObjects* in the context menu for the folders for dimensions or key figures. In the dialog box that appears, you can enter and transfer up to ten InfoObjects directly, or you can select them using input help. You can use drag and drop to reassign them.

5. Details and object-specific properties:

   If you double click on an InfoObject, the detail display of this InfoObject is displayed. In the context menu of an InfoObject, you can make additional settings under *Object-Specific Properties*. See the section *Structure-Specific Properties of InfoObject* in Additional Functions in InfoCube Maintenance [Page 133].

6. Create dimensions: The dimensions data package, time, and unit are available as the default setting. The data package dimension contains technical characteristics. Time characteristics and units are automatically assigned to the corresponding dimensions. In the context menu of the *Dimensions* folder, you can create additional dimensions under *Create New Dimensions*.

   For more information, see Dimension [Page 128].

   If a dimension only has one characteristic, or it has a large number of values, you need to set the *Line Item* or *High Cardinality* indicator. For more information, see Line Item and High Cardinality [Page 129].

7. In the context menu of the *Key Figures* folder you can *Insert New Hierarchy Nodes*. In this way you can sort the key figures in a hierarchy. You then get a better overview of large quantities of key figures in query definition.

   See also: Defining New Queries [External]

8. *Save* or *Activate* the InfoCube.

   Only an activated InfoCube can be supplied with data and used for reporting and analysis.

Next Step

Creating Transformations [Page 242]

4.1.6.7.1.4 eal-Time InfoCubes

Definition

Real-time InfoCubes differ from standard InfoCubes in their ability to support parallel write accesses. Standard InfoCubes are technically optimized for read accesses to the detriment of write accesses.

Use

Real-time InfoCubes are used in connection with the entry of planning data. For more information, see:

- BI Integrated Planning: InfoProvider [External]
- Overview of Planning with BW-BPS [External]

The data is simultaneously written to the InfoCube by multiple users. Standard InfoCubes are not suitable for this. You should use standard InfoCubes for read-only access (for example, when reading reference data).

Structure

Real-time InfoCubes can be filled with data using two different methods: Using the transaction for entering planning data and using BI staging, whereas planning data then cannot be loaded simultaneously. You have the option to convert a real-time InfoCube. Select *Convert Real-Time InfoCube* using the context menu in your real-time InfoCube in the InfoProvider tree. By default, *Real-Time Cube Can Be Planned, Data Loading Not Permitted* is selected. Switch this setting to *Real-Time Cube Can Be Loaded With Data; Planning Not Permitted* if you want to fill the cube with data using BI Staging.

During entry of planning data, the data is written to a real-time InfoCube data request. As soon as the number of records in a data request exceeds a threshold value, the request is closed and a rollup is carried out for this request in defined aggregates (asynchronously). You can still rollup and define aggregates, collapse, and so on, as before.

Depending on the database on which they are based, real-time InfoCubes differ from standard BasisCubes in the way they are indexed and partitioned. For an Oracle DBMS this means, for example, no Bitmap indexes for the fact table and no partitioning (initiated by BI) of the fact table according to the packet dimensions.

Reduced read-only performance is accepted as a drawback of real-time InfoCubes, in favor of the option of parallel (transactional) writing and improved write performance.

Creating a Real-Time InfoCube

When creating a new InfoCube in the Data Warehousing Workbench, select the *Real-Time* indicator.

Converting a Standard InfoCube into a Real-Time InfoCube

Conversion with Loss of Transaction Data

If the standard InfoCube already contains transaction data but you no longer need it (for example, test data from the implementation phase of the system), you proceed as follows:

1. In the InfoCube maintenance in the Data Warehousing Workbench, from the main menu, choose *InfoCube* → *Delete Data Content.* The transaction data is deleted and the InfoCube is set to *inactive*.
2. Continue with the same procedure as with creating a real-time InfoCube.

Conversion with the Retention of Transaction Data

If the standard InfoCube already contains transaction data from the production operation that you still need, proceed as follows:

Execute the SAP_CONVERT_TO_TRANSACTIONAL ABAP report under the name of the corresponding InfoCube. You should schedule this report as a background job for InfoCubes with more than 10,000 data records. This is to avoid a potentially long runtime.

Integration

The following typical scenarios arise for the use of real-time InfoCubes in planning:

1. Scenario:

Actual data (read-only access) and planned data (read-only and write access) have to be held in different InfoCubes. Therefore, use a standard InfoCube for actual data and a real-time InfoCube for planned data. Data integration is achieved using a multiplanning area that contains the areas that are assigned to the InfoCubes. Access to the two different InfoCubes is controlled by the *Planning area* characteristic that is automatically added.

2. Scenario:

In this scenario, the planned and actual data have to be together in one InfoCube. This is the case, for example, with special rolling forecast variants. You have to use a real-time InfoCube, since both read-only and write accesses take place. You can no longer load data directly that has already arrived in the InfoCube by means of an upload or import source. To be able to load data nevertheless, you have to make a copy of the real-time InfoCube that is identified as a standard InfoCube and not as real-time. Data is loaded as usual and subsequently updated to the real-time InfoCube.

4.1.6.7.1.5 Additional Functions in InfoCube Maintenance

Documents

This function allows you to display, create or change documents for InfoCubes.

See also: Documents [External].

Tree Display

Use this function to display, in a clear tree structure, all the settings for an InfoCube that have been made in InfoCube maintenance. The InfoCube is output in a **hierarchical tree display** with its dimensions and InfoObjects.

Version Comparison

You can use this function to compare the following InfoCube versions with regard to changes in the InfoCube maintenance:

- the active and modified versions of an InfoCube
- the active and Content versions of an InfoCube
- the modified and Content versions of an InfoCube

Transport Connection

You can choose and transport InfoCubes. All those BW objects that are needed to ensure a consistent status in the target system are collected automatically.

Where-Used List

You can determine which other objects in BW can use a specific InfoCube.

You can also determine the consequences of making a particular change to an InfoCube and whether the change is permitted at the moment.

Business Content

In InfoCubes from Business Content, you can use this function to jump to the transaction to transfer the business content, to copy the InfoCube and to compare the InfoCube with the customer version. For more information, see Business Content (Versions) [External].

Navigational Attributes

A navigation attribute [Page 90] that has been created in the InfoObject maintenance for a characteristic can be switched on in the InfoCube. With the default setting, they are switched off in order to implicitly include as few attributes as possible in the InfoCube. See also Performance of Navigation Attributes in Queries and Input Help [Page 92].

Note that you can subsequently create or activate navigation attributes in the InfoCube at any time. However, once you have activated an attribute, you are no longer able to deactivate it (because of any aggregates or selection variables that may have been defined).

Units

By clicking a pushbutton on the key figures screen, you can display the units contained in the InfoCube. Units are not defined, but result from data from the transferred key figures.

Analyzing InfoCubes

You get to the analysis and repair environment by choosing *Edit* from the main menu. You can use the analysis and repair environment to check the consistency of your InfoCube.

See Analysis and Repair Environment [Page 792]

Structure-Specific Properties of the InfoObject

Using *Extras* in the main menu, you can assign InfoObjects specific properties that are valid only in the InfoCube you are currently processing.

These settings correspond for the most part to the settings that you can make globally for an InfoObject. For characteristics, these are *Display, Text Type*, *Selection* and *Filter Value Selection upon Query Execution.* See also the corresponding section in Tab Page: Business Explorer [Page 63].

You are also able to specify constants for characteristics:

A fixed value is assigned to a characteristic when you assign a constant. This means that the characteristic is available on the database (for verifications, for example) but is no longer displayed in the query (no aggregation / drilldown is possible for this characteristic).

It is particularly useful to assign constants for compound characteristics.

Example 1:

The *storage location* characteristic is compounded with the *plant* characteristic. If only one plant is ever run within the application, you can assign a constant to the plant. The verification for the storage-location master table runs correctly with this value for the plant. In the query, however, only the storage location appears as a characteristic.

Example 2:

For an InfoProvider, you make the setting that only the constant 2005 appears for the year. In a query on a MultiProvider that contains this InfoProvider, the InfoProvider is disregarded if the selection is for year 2004. Query performance

is improved in this way as the system knows from the outset that it does not have to search for any records.

Special case:

If the constant SPACE (type CHAR) or 00..0 (type NUMC) is assigned to the characteristic, specify the character # in the first position.

For key figures, the settings *Decimal Places* and *Display* are available. See also the corresponding section in Tab Page: Additional Properties [Page 110].

Info Functions

Various information functions are available referring to the status of the InfoCube:

- The log display for the save, activation and deletion runs of the InfoCube,
- The InfoCube status in the ABAP/4 Dictionary and on the database,
- a function for displaying the raw data (browser) for the data saved in the InfoCube [Page 136]
- The current system settings,
- The permitted limits in the InfoCube,
- The object catalog entry.
- Analysis of data consistency of the InfoCube [External]

Special Functions

Navigation to *InfoObject Maintenance:* Pushbuttons allow you to *create, display,* and *change* individual InfoObjects. Note that changes to InfoObjects apply to all the places where the InfoObject is used (that means also in other InfoCubes)!

Undo change: This function resets the InfoCube to the active version, meaning that the changes that were made the last time data was saved, are undone.

Display active / SAP version: When you are editing the InfoCube, you can display its active version, or the version delivered by SAP (if it exists).

Performance Settings:

- DB Save Parameters [Page 659]
- Partitioning [Page 660]
- Non-Cumulative Parameter [Page 136]

Assigning a Function Module

You load data from external sources by assigning a function module to an InfoCube. The function module is called when the data is loaded. It supplies the data temporarily. The function can be called from the context menu using *Additional Characteristics*.

4.1.6.7.1.5.1 Checking the Data Loaded in the InfoCube

Prerequisites

You have loaded your data into the InfoCube, and checked the data request in the Monitor [External].

Procedure

Transaction Data:

1. Choose *InfoCube Maintenance* → *Edit* → *InfoCube Data Display,* and specify whether you want to include the SIDs in the display as well.
2. Choose the characteristic values for which the output list is to be selected.
3. Choose *field selection for output*, and select the characteristics and key figures that are to be selected in the output list.
4. Choose *Execute*.
5. In the following window, choose *Execute* again.

See also: InfoCube Content [External]

Master data:

1. Choose *InfoSource Tree* → *Your InfoArea* → *Your Master Data InfoSource* → *Context Menu (right mouse button)* → *Change Attributes.*
2. Select the *Master Data/Texts* folder.
3. Double click on the technical name of the master data table.
4. In the following window, choose *Utilities* → *Table Contents* → *Display*.

4.1.6.7.1.5.2 Non-Cumulative Value Parameter Maintenance

Use

The non-cumulative parameter maintenance is activated as soon as at least one non-cumulative value exists in the InfoCube. In this case, you have to choose a **time reference characteristic for the non-cumulative key figures** of the InfoCube, which sets the granularity (the degree of precision) in which the non-cumulative values are managed. This time reference characteristic applies for all the non-cumulative values of the InfoCube and must be the “smallest” of the time characteristics present in the InfoCube.

See also: Time Reference Characteristics [Page 422]

The InfoCube contains warehouse stock key figures as well as the time characteristics ‘calendar month’ and ‘calendar year’. In this case, define the InfoObject 0CALMONTH (calendar month) as a reference characteristic for the time-based aggregation.

For the non-cumulative values contained in the InfoCube, a **validity table** is created, in which the time interval is stored, for which the non-cumulative values

are valid. Apart from the reference characteristic for time-based aggregation, which is always implicitly inserted in the validity table, this table can contain other additional characteristics. See also: Validity Area [Page 426].

Such a characteristic is, for example, the characteristic “Plant”, if the non-cumulative key figures for different times are reported, for example, from different source systems.

For plan/actual values, no validity area (plan values until year end, actual values only until current date) needs to be maintained. Instead, you need to create a MultiProvider for these types of scenarios.

Activities

Define all additional characteristics that should be contained in the validity table, by selecting. In the aforementioned example, the characteristics “plan/actual” and “plant” must be selected. The system automatically generates the validity table corresponding to the definition that was made. This table is updated automatically when loading data.

See also:

Modeling Non-Cumulatives with Non-Cumulative Key Figures [Page 420]

4.1.6.7.1.5.3 DB Storage Parameters

Use

You can maintain database storage parameters for PSA tables, master data tables, InfoCube fact- and dimension tables, and DataStore object tables.

Use this setting to determine how the system handles the table when it creates it in the database:

- Use *Data Type* to set in which physical database area (tablespace) the system is to create the table. Each data type (master data, transaction data, organization- and Customizing data, customer data) has its own physical database area, in which all tables assigned to this data type are stored. If selected correctly, your table is automatically assigned to the correct area when it is created in the database.

  We recommend you use separate tablespaces for very large tables.

  You can find information about creating a new data type in SAP Note 0046272 (Introduce new data type in technical settings).

- Via *Size Category*, you can set the amount of space the table is thought to need in the database. Five categories are available in the input help. You can also see here how many data records correspond to each individual category. When creating the table, the system reserves an initial storage space in the database. If the table later requires more storage space, it obtains it as set out in the size category. Correctly setting the size category prevents there being too many small extents (save areas) for a table. It also prevents the wastage of storage space when creating extents that are too large.

You can use the maintenance for storage parameters to better manage databases that support this concept.

You can find additional information about the data type and size category parameters in the ABAP Dictionary table documentation, under Technical Settings [External].

PSA Table

For PSA tables, you access the database storage parameter maintenance by choosing *Goto → Technical Attributes* in DataSource maintenance. In dataflow 3.x, you access this setting in transfer rule maintenance in the *Extras* menu.

You can also assign storage parameters for a PSA table already in the system. However, this has no effect on the existing table. If the system generates a new PSA version (a new PSA table) due to changes to the DataSource, this is created in the data area for the current storage parameters.

InfoObject Tables

For InfoObject tables, the maintenance of database storage parameters can be found in the InfoObject maintenance, in the *Extras* menu.

InfoCube/aggregate fact and dimension tables

For fact- and dimension tables, the maintenance of database storage parameters can be found in the InfoCube maintenance, in the *Extras* menu.

DataStore Object Tables (Activation Queue and Table for Active Data)

For DataStore object tables, the maintenance of database storage parameters can be found in DataStore object maintenance, in the *Extras* menu.

4.1.6.7.1.5.4 Partitioning

Use

You use partitioning to split the total dataset for an InfoProvider into several, smaller, physically independent and redundancy-free units. This separation improves system performance when you analyze data delete data from the InfoProvider.

Integration

All database providers except DB2 for Linux, UNIX, and Windows support partitioning. You can use clustering to improve the performance for DB2 for Linux, UNIX, and Windows.

If you are using IBM DB2 for i5/OS as the DB platform, you require database version V5R3M0 or higher and an installation of component DB2 Multi System. Note that with this system constellation the BI system with active partitioning can only be copied to other IBM iSeries with an SAVLIB/RSTLIB operation (homogeneous system copy). If you are using this database you can also partition PSA tables. You first have to activate this function using RSADMIN parameter DB4_PSA_PARTITIONING = 'X'. SAP Note 815186 includes more comprehensive information on this.

Prerequisites

You can only partition a dataset using one of the two partitioning criteria ‘calendar month’ (0CALMONTH) or ‘fiscal year/period (0FISCPER). At least one of the two InfoObjects must be contained in the InfoProvider.

If you want to partition an InfoCube using the fiscal year/period (0FISCPER) characteristic, you have to set the fiscal year variant characteristic to constant.

See Partitioning InfoCubes using Characteristic 0FISCPER [Page 140].

Features

When you activate the InfoProvider, the system creates the table on the database with one of the number of partitions corresponding to the value range. You can set the value range yourself.

Choose the partitioning criterion 0CALMONTH and determine the value range

From 01.1998

To 12.2003

6 years x 12 months + 2 = 74 partitions are created (2 partitions for values that lay outside of the range, meaning < 01.1998 or >12.2003).

You can also determine the maximum number of partitions created on the database for this table.

Choose the partitioning criterion 0CALMONTH and determine the value range

From 01.1998

To 12.2003

Choose 30 as the maximum number of partitions.

Resulting from the value range: 6 years x 12 calendar months + 2 marginal partitions (up to 01.1998, from 12.2003) = 74 single values.

The system groups three months together at a time in a partition (meaning that a partition corresponds to exactly one quarter); in this way, 6 years x 4 partitions/year + 2 marginal partitions = 26 partitions created on the database.

The performance gain is only achieved for the partitioned InfoProvider if the time characteristics of the InfoProvider are consistent. This means that with a partition using 0CALMONTH, all values of the 0CAL x characteristics of a data record have to match.

In the following example, only record 1 is consistent. Records 2 and 3 are not consistent:

0CALYEAR	0CALMONTH	0CALDAY
1998	05.1998	28.05.1998
1998	00.0000	00.00.0000
1998	05.1998	01.01.1998

Note that you can only change the value range when the InfoProvider does not contain data. If data has already been loaded to the InfoProvider, you have to perform repartitioning.

For more information, see Repartitioning [Page 141].

We recommend that you use “partition on demand“. This means that you should not create partitions that are too large or too small. If you choose a time period that is too small, the partitions are too large. If you choose a time period that ranges too far into the future, the number of partitions is too great. Therefore we recommend that you create a partition for a year, for example, and that you repartition the InfoProvider after this time.

Activities

In InfoProvider maintenance, choose *Extras* → *Partitioning* and specify the value range. Where necessary, limit the maximum number of partitions.

4.1.6.7.1.5.4.1 Partitioning InfoCubes Using the Characteristic 0FISCPER

Use

You can partition InfoCubes using two characteristics – calendar month (0CALMONTH) and fiscal year/period (0FISCPER). The special feature of the fiscal year/period characteristic (0FISCPER) being compounded with the fiscal year variant (0FISCVARNT) means that you have to use a special procedure when you partition an InfoCube using 0FISCPER.

Prerequisites

When partitioning using 0FISCPER values, values are calculated within the partitioning interval that you specified in the InfoCube maintenance. To do this, the value for 0FISCVARNT must be known at the time of partitioning; it must be set to **constant**.

Procedure

1. The InfoCube maintenance is displayed. Set the value for the 0FISCVARNT characteristic to constant. Carry out the following steps:
   a. Choose the *Time Characteristics* tab page.
   b. In the context menu of the dimension folder, choose *Object specific InfoObject properties.*
   c. Specify a constant for the characteristic 0FISCVARNT. Choose *Continue*.
2. Choose *Extras* → *DB Performance* → *Partitioning.* The *Determine Partitioning Conditions* dialog box appears. You can now select the 0FISCPER characteristic under *Slctn*. Choose *Continue*.
3. The *Value Range (Partitioning Condition)* dialog box appears. Enter the required data.
4. For more information, see Partitioning [Page 660].

4.1.6.7.1.5.5 Repartitioning

Use

Repartitioning can be useful if you have already loaded data to your InfoCube, and:

- You did not partition the InfoCube when you created it.
- You loaded more data into your InfoCube than you had planned when you partitioned it.
- You did not choose a long enough period of time for partitioning.
- Some partitions contain no data or little data due to data archiving over a period of time.

Integration

All database providers support this function except DB2 for Linux, UNIX, Windows and MAXDB. For DB2 for Linux, UNIX and Windows, you can use clustering or reclustering instead. For more information, see Clustering [Page 662] .

Features

Merging and Adding Partitions

When you *merge and add partitions,* InfoCube partitions are either merged at the bottom end of the partitioning schema (merge), or added at the top (split).

Ideally, this operation is only executed for the database catalog. This is the case if all the partitions that you want to merge are empty and no data has been loaded outside of the time period you initially defined. The runtime of the action is only a few minutes.

If there is still data in the partitions you want to merge, or if data has been loaded beyond the time period you initially defined, the system saves the data in a shadow table and then copies it back to the original table. The runtime depends on the amount of data to be copied.

With InfoCubes for non-cumulatives, all markers are either in the bottom partition or top partition of the E fact table. Whether mass data also has to be copied depends on the editing options. For this reason, the partitions of non-cumulative InfoCubes cannot be merged if all of the markers are in the bottom partition. If all of the markers are in the top partition, adding partitions is not permitted. If this is the case, use the *Complete Repartitioning* editing option.

You can merge and add partitions for aggregates as well as for InfoCubes. Alternatively, you can reactivate all of the aggregates after you have changed the InfoCube. Since this function only changes the DB memory parameters of fact tables, you can continue to use the available aggregates without having to modify them.

We recommend that you completely back up the database before you execute this function. This ensures that if an error occurs (for example, during a DB catalog operation), the can restore the system to its previous status.

Complete Partitioning

Complete Partitioning fully converts the fact tables of the InfoCube. The system creates shadow tables with the new partitioning schema and copies all of the data from the original tables into the shadow tables. As soon as the data is copied, the system creates indexes and the original table replaces the shadow table. After the system has successfully completed the

partitioning request, both fact tables exist in the original state (shadow table), as well as in the modified state with the new partitioning schema (original table). You can manually delete the shadow tables after repartitioning has been successfully completed to free up the memory. Shadow tables have the namespace `/BIC/4F<Name of InfoCube>` or `/BIC/4E<Name of InfoCube>`.

You can only use complete repartitioning for InfoCubes. A heterogeneous state is possible. For example, it is possible to have a partitioned InfoCube with non partitioned aggregates. This does not have an adverse effect on functionality. You can automatically modify all of the active aggregates by reactivating them.

Monitor

You can monitor the repartitioning requests using a monitor. The monitor shows you the current status of the processing steps. When you double-click, the relevant logs appear. The following functions are available in the context menu of the request or editing step:

- *Delete:* You delete the repartitioning request; it no longer appears in the monitor, and you cannot restart. All tables remain in their current state. The InfoCube may be inconsistent.
- *Reset Request:* You reset the repartitioning request. This deletes all the locks for the InfoCube and all its shadow tables.
- *Reset Step:* You reset the canceled editing steps so that they are reset to their original state.
- *Restart:* You restart the repartitioning request in the background. You cannot restart a repartitioning request if it still has status *Active* (yellow) in the monitor. Check whether the request is still active (transaction SM37) and, if necessary, reset the current editing step before you restart.

Background Information About Copying Data

By default, the system copies a maximum of six processes in parallel. The main process splits dialog processes in the background. These dialog processes each copy small data packages and finish with a COMMIT. If a timeout causes one of these dialog processes to terminate, you can restart the affected copy operations, after you have altered the timeout time. To do this, choose *Restart Repartitioning Request.*

Background Information About Error Handling

Even if you can restart the individual editing steps, you should not reset the repartitioning request or the individual editing steps without first performing an error analysis.

During repartitioning, the relevant InfoCube and its aggregates are locked against modifying operations (loading data, aggregation, rollup and so on) to avoid inconsistent data. In the initial dialog, you can manually unlock objects. This option is only intended for cases where errors have occurred and should only be used after the logs and datasets have been analyzed.

Transport

Since the metadata in the target system is adjusted without the DB tables being converted when you transport InfoCubes, repartitioned InfoCubes may only be transported when the repartitioning has already taken place in the target system. Otherwise inconsistencies that can only be corrected manually occur in the target system.

Activities

You can access repartitioning in the Data Warehousing Workbench using *Administration*, or in the context menu of your InfoCube.

You can schedule repartitioning in the background by choosing *Initialize.* You can monitor the repartitioning requests by choosing *Monitor*.

4.1.6.7.1.5.6 Clustering

Use

Clustering allows you to save sorted data records in the fact table of an InfoCube. Data records with the same dimension keys are saved in the same extents (related database storage unit). This means that same data records are not spread across a large memory area and thereby reduces the number of extents that the system has to read when it accesses tables. This greatly accelerates read, write and delete access to the fact table.

Prerequisites

Currently the function is only supported by the database platform DB2 for Linux, UNIX, and Windows. You can use partitioning to improve the performance of other databases. For more information, see Partitioning [Page 660].

Features

Two types of clustering are available: Index clustering and multidimensional clustering (MDC).

Index Clustering

Index clustering organizes the data records of a fact table according to the sort sequence of an index. Organization is linear and corresponds to the values of the index field.

If a data record cannot be inserted in accordance with the sort sequence because the relevant extent is already full, the data record is inserted into an empty extent at the end of the table. For this reason, the system cannot guarantee that the sort sequence is always correct, particularly if you perform many insert and delete operations. Reorganizing the table restores the sort sequence and frees up memory space that is no longer required.

The clustering index of an F fact table is, by default, the secondary index in the time dimension. The clustering index of an E fact table is, by default, the acting primary index (P index).

As of release SAP BW 2.0, index clustering is standard for all InfoCubes and aggregates.

Multidimensional Clustering (MDC)

Multidimensional clustering organizes the data records of a fact table in accordance with one or more fields that you define freely. The selected fields are also marked as MDC dimensions. Only data records that have the same values in the MDC dimensions are saved in an extent. In the context of MDC, an extent is called a block. The system can always guarantee that the sort sequence is correct. Reorganizing the table is not necessary, even with many insert and delete operations.

Block indexes from within the database, instead of the default secondary indexes, are created for the selected fields. Block indexes link to extents instead of data record numbers and are

therefore much smaller. They save memory space and the system can search through them more quickly. This accelerates table requests that are restricted to these fields.

You can select the key fields of the time dimension or any customer-defined dimensions of an InfoCube as an MDC dimension. You cannot select the key field of the package dimension; it is automatically added to the MDC dimensions in the F fact table.

You can also select a time characteristic instead of the time dimension. In this case, the fact table has an extra field. This contains the SID values of the time characteristic. Currently only the time characteristics *Calendar Month* (0CALMONTH) and *Fiscal Year/Period* (0FISCPER) are supported. The time characteristic must be contained in the InfoCube. If you select the *Fiscal Year/Period* (0FISCPER) characteristic, a constant must be set for the *Fiscal Year Variant* (0FISCVARNT) characteristic.

Clustering is applied to all the aggregates of the InfoCube. If an aggregate does not contain an MDC dimension of the InfoCube, or if all the InfoObjects of an MDC dimension are created as line item dimensions in the aggregate, the aggregates are clustered using the remaining MDC dimensions. Index clustering is used for the aggregate if the aggregate does not contain any MDC dimensions of the InfoCube, or if it only contains MDC dimensions.

Multidimensional clustering was introduced in Release SAP NetWeaver 2004s and can be set up separately for each InfoCube.

For procedures, see Definition of Clustering [Page 144].

4.1.6.7.1.5.6.1 Definition of Clustering

Prerequisites

Note: You can only change the MDC dimensions if the InfoCube does not contain any data. If data has already been loaded you must perform *Reclustering.*

For more information, see Reclustering [Page 146].

Features

In InfoCube maintenance, select *Extras* → *DB Performance* → *Clustering* and specify the MDC dimensions.

Selecting Clustering

You can choose between *Index Clustering* and *Multidimensional Clustering* on the selecting clustering screen.

Multidimensional Clustering

You can select MDC dimensions for the InfoCube on the *Multidimensional Clustering* screen.

Under *Time Dimension*, under a selected column, you can select a time dimension field as an MDC dimension. As long as they are contained in the InfoCube, you can select either the key field of the time dimension, the additional SID field of the *Calendar Month* (0CALMONTH) time characteristic, the additional SID field of the *Fiscal Year/Period* (0FISCPER) time characteristic or, if you do not want to select the time dimension as an MDC dimension, no field at all.

The system automatically assigns sequence number 1 to the time dimension field. The sequence number shows whether a field has been selected as an MDC dimension, and determines the order of the MDC dimensions in the combined block index.

In addition to block indexes for the different MDC dimensions within the database, the system creates the combined block index. The combined block index contains the fields of all the MDC dimensions. The order of the MDC dimensions can slightly affect the performance of table queries that are restricted to all MDC dimensions and those that are used to access the combined block index.

Under *Characteristic Dimensions* you can select additional MDC dimensions and assign them consecutive sequence numbers. You can select the key fields of the unit dimension and the key fields of all customer dimensions, as long as they contain characteristics.

For more information about selecting dimensions, see Selecting MDC Dimensions [Page 145].

4.1.6.7.1.5.6.2 Selecting MDC Dimensions

When selecting MDC dimensions, proceed as follows:

- Select dimensions for which you often use restrictions in queries.
- Select dimensions with a low cardinality.

  The MDC dimension is created in the column with the dimension keys (DIMID). The number of different combinations in the dimension characteristics determines the cardinality. Therefore, select a dimension with either one, or few characteristics and with only a few different characteristic values.

  Line item dimensions are not usually suitable, as they normally have a characteristic with a high cardinality.

  If you specifically want to create an MDC dimension for a characteristic with a low cardinality, you can define this characteristic as a line item dimension in the InfoCube. This differs from the norm that line item dimensions contain characteristics with a very high cardinality. However, this has the advantage for multidimensional clustering that the fact table contains the SID values of the characteristic, in place of the dimension keys, and the database query can be restricted to these SID values.
- You cannot select more than three dimensions, including the time dimension.
- Assign sequence numbers, using the following criteria:
  - Sort the dimensions according to how often they occur in queries (assign the lowest sequence number to the InfoObject that occurs most often in queries).
  - Sort the dimensions according to selectivity (assign the lowest sequence number to the dimension with the most different data records).

    Note: At least one block is created for each value combination in the MDC dimension. This memory area is reserved independently of the number of data records that have the same value combination in the MDC dimension. If there is not a sufficient number of data records with the same value combinations to completely fill a block, the free memory remains unused. This is so that data

records with a different value combination in the MDC dimension cannot be written to the block.

If for each combination that exists in the InfoCube, only a few data records exist in the selected MDC dimension, most blocks have unused free memory. This means that the fact tables use an unnecessarily large amount of memory space. Performance of table queries also deteriorates, as many pages with not much information must be read.

Example

The size of a block depends on the PAGESIZE and the EXTENTSIZE of the tablespace. The standard PAGESIZE of the fact-table tablespace with the assigned data class DFACT is 16K. Up to Release SAP BW 3.5, the default EXTENTSIZE value was 16. As of Release SAP NetWeaver 2004s the new default EXTENTSIZE value is 2.

With an EXTENTSIZE of 2 and a PAGESIZE of 16K the memory area is calculated as 2 x 16K = 32K, this is reserved for each block.

The width of a data record depends on the number of dimensions and the number of key figures in the InfoCube. A dimension key field uses 4 bytes and a decimal key figure uses 9 bytes. If, for example an InfoCube has 3 standard dimensions, 7 customer dimensions and 30 decimal key figures, a data record needs 10 x 4 bytes + 30 x 9 bytes = 310 bytes. In a 32K block, 32768 bytes / 310 bytes could write 105 data records.

If the time characteristic calendar month (0CALMONTH) and a customer dimension are selected as the MDC dimension for this InfoCube, at least 100 data records should exist for each InfoPackage, for each calendar month and for each dimension key of the customer dimension. This allows optimal use of the memory space in the F fact table. In the E fact table, this is valid for each calendar month and each dimension key of the customer dimension.

4.1.6.7.1.5.7 Reclustering

Use

Reclustering allows you to change the clustering of InfoCubes and DataStore objects that already contain data. You may need to make a correction if, for example, there are only a few data records for each of the value combinations of the selected MDC dimension and as a result the table uses an excessive amount of memory space. To improve the performance of database queries, you may want to introduce multidimensional clustering for InfoCubes or DataStore objects.

Integration

This function is only available for the database platform DB2 for Linux, UNIX, and Windows. You can use partitioning to improve the performance of other databases. For more information, see Partitioning [Page 660].

Features

Reclustering InfoCubes

With reclustering, the InfoCube fact tables are always completely converted. The system creates shadow tables with a new clustering schema and copies all of the data from the

original tables into the shadow tables. As soon as the data is copied, the system creates indexes and the original table replaces the shadow table. After the reclustering request has been successfully completed, both fact tables exist in their original state (name of shadow table) as well as in their modified state with the new clustering schema (name of original table).

You can only use reclustering for InfoCubes. Reclustering deactivates the active aggregates of the InfoCubes; they are reactivated after the conversion.

Reclustering DataStore Objects

Reclustering completely converts the active table of the DataStore object. The system creates a shadow table with a new clustering schema and copies all of the data from the original table into the shadow table. As soon as the data is copied, the system creates indexes and the original table replaces the shadow table. After the reclustering request has been successfully completed, both active tables exist in their original state (name of shadow table) as well as in their modified state with the new clustering schema (name of original table).

You can only use reclustering for standard DataStore objects and DataStore objects for direct update. You cannot use reclustering for write-optimized DataStore objects. User-defined multidimensional clustering is not available for write-optimized DataStore objects.

Monitoring

You can monitor the clustering request using a monitor. The monitor shows you the current status of the processing steps. When you double-click, the relevant logs appear. The following functions are available in the context menu of the request or editing step:

- *Delete:* You delete the clustering request. It no longer appears in the monitor and you cannot restart. All tables remain in their current state. This may result in inconsistencies in the InfoCube or DataStore object.
- *Reset Request:* You reset the clustering request. This deletes all the locks for the InfoCube and all its shadow tables.
- *Reset Step:* You reset the canceled editing steps so that they are reset to their original state.
- *Restart:* You restart the clustering request in the background.

Background Information About Copying Data

By default, the system copies a maximum of six processes in parallel. The main process splits dialog processes in the background. These dialog processes each copy small data packages and finish with a COMMIT. If a timeout causes one of these dialog processes to terminate, you can restart the affected copy operations after you have altered the timeout time. To do this, choose *Restart Reclustering Request.*

Activities

You access reclustering in the Data Warehousing Workbench under *Administration* or in the context menu of your InfoCube or DataStore object.

You can schedule repartitioning in the background by choosing *Initialize.* You can monitor the clustering requests by choosing *Monitor*.

4.1.6.7.1.6 Overview of Loadable InfoSources for an InfoCube

Usage

In the InfoCube tree of the Administrator Workbench Modeling, you can display all InfoSources for an InfoCube to which it is possible to load data.

Activities

1. Select the InfoCube and choose *InfoSources Overview* using the context menu (right mouse button).

   Information on the InfoSource as well as on the last loading process is displayed for you.

2. Using the status symbol of the last loading process you get to the Monitor and can check this data request.
3. Using the pushbutton *Expand* you get to the InfoSource tree. From here you can schedule a data request for the InfoCube.

4.1.6.7.2 DataStore Objects

Definition

A DataStore object serves as a storage location for consolidated and cleansed transaction data or master data on a document (atomic) level.

This data can be evaluated using a BEx query.

A DataStore object contains key fields (for example, document number/item) and data fields that can also contain character fields (for example, order status, customer) as key figures. The data from a DataStore object can be updated with a delta update into InfoCubes and/or other DataStore objects or master data tables (attributes or texts) in the same system or across different systems.

Unlike multidimensional data storage using InfoCubes, the data in DataStore objects is stored in transparent, flat database tables. The system does not create fact tables or dimension tables.

Use

The cumulative update of key figures is supported for DataStore objects, just as it is with InfoCubes, but with DataStore objects it is also possible to overwrite data fields. This is particularly important with document-related structures. If documents are changed in the source system, these changes include both numeric fields, such as the order quantity, and non-numeric fields, such as the ship-to party, status and delivery date. To reproduce these changes in the DataStore objects in the BI system, you have to overwrite the relevant fields in the DataStore objects and set them to the current value. Furthermore, you can use an overwrite and the existing change log to render a source delta enabled. This means that the delta that is further updated to the InfoCubes, for example, is calculated from two successive after-images.

There are different types of DataStore object:

- Standard (see Standard DataStore Objects [Page 151])
- For direct update (see DataStore Objects for Direct Update [Page 152])

- Write-optimized (see Write-Optimized DataStore Objects [Page 154])

Differences between DataStore object types:

Type	Structure	Data Supply	SID Generation	Are BEx Queries Possible?
Standard	Consists of three tables: activation queue, table of active data, change log	From data transfer process	Yes	Yes
For direct update	Consists of the table of active data only	From APIs	No	Yes
Write-optimized	Consists of the table of active data only	From data transfer process	No	Yes

Integration

Integration with the Data Warehousing Workbench - Modeling

Metadata

DataStore objects are fully integrated with BI metadata. They are transported just like InfoCubes and are installed from BI Content (for more information, see Installing Business Content [Page 435]). DataStore objects are grouped with InfoCubes in the InfoProvider view of the Data Warehousing Workbench - Modeling, and are displayed in a tree. They also appear in the data flow display.

Update

Transformation rules define the rules that are used to write data to a DataStore object. They are very similar to the transformation rules for InfoCubes. The main difference is the behavior of data fields in the update. When you update requests into a DataStore object, you have an overwrite option as well as an addition option.

See also Transformation Type [Page 240].

The delta process [Page 534] that is defined for the DataSource also influences the update. When loading files, the user must select a suitable delta process so that the correct transformation type is used.

Unit fields and currency fields operate just like normal key figures, meaning that they must be explicitly filled using a rule.

Scheduling and Monitoring

The processes for scheduling the data transfer process for updating data into InfoCubes and DataStore objects are identical.

It is also possible to schedule the activation of DataStore object data and the update from the DataStore object into the related InfoCubes or DataStore objects.

The individual steps, including processing the DataStore object, are logged in the monitor.

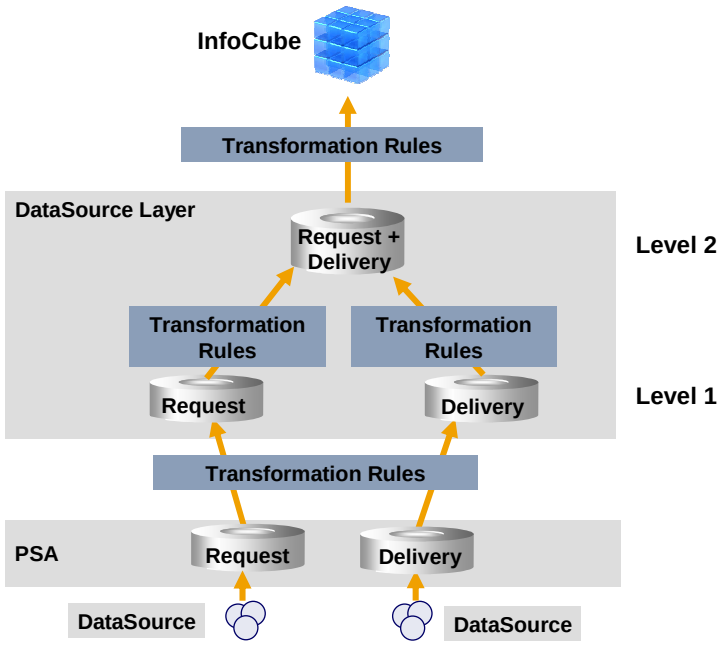
There is a separate detailed monitor for executed request operations (such as activation or rollback).

Loadable DataSources

In full-update mode, each transaction data DataSource contained in a DataStore object can be updated. In delta-update mode, only those DataSources that are flagged as delta-enabled DataStores can be updated.

4.1.6.7.2.1 Scenario for DataStore Objects

The diagram below shows how DataStore objects are used in this example of updating order and delivery information, and the status tracking of orders, meaning which orders are open, which are partially-delivered, and so on.



There are three main steps to the entire data process:

Loading the data into the BI system and storing it in the PSA [Page 46]

The data requested by the BI system is stored initially in the PSA. A PSA is created for each DataSource and each source system. The PSA is the storage location for incoming data in the BI system. Requested data is saved, unchanged, to the source system.

1. Processing and storing the data in DataSource objects

   In the second step, the DataSource objects are used on two different levels.

   a. On level one, the data from multiple source systems is stored in DataSource objects. Transformation rules permit you to store the consolidated and cleaned up data in the technical format of the BI system. On level one, the data is stored on the document level (for example, orders and deliveries) and constitutes the consolidated database for further processing in the BI system. Data analysis is therefore not usually performed on the DataSource objects at this level.

   b. On level two, transfer rules subsequently combine the data from several DataStore objects into a single DataStore object in accordance with business-

related criteria. The data is very detailed, for example, information such as the delivery quantity, the delivery delay in days, and the order status, are calculated and stored per order item. Level 2 is used specifically for operative analysis issues, for example, which orders are still open from the last week. Unlike multidimensional analysis, where very large quantities of data are selected, here data is displayed and analyzed selectively.

2. Storing data in the InfoCube

   In the final step, the data is aggregated from the DataStore object on level two into an InfoCube, meaning in this scenario, that the InfoCube does not contain the order number, but saves the data, for example, on the levels of *customer, product*, and *month*. Multidimensional analysis is also performed on this data using a BEx query. You can still display the detailed document data from the DataStore object whenever you need to. Use the report/report interface from a BEx query. In this way, you are able to analyze the aggregated data from the InfoCube, and target the specific level of detail you want to access in the data.

4.1.6.7.2.2 Standard DataStore Object

Use

The standard DataStore object is filled with data during the extraction and load process in the BI system.

Structure

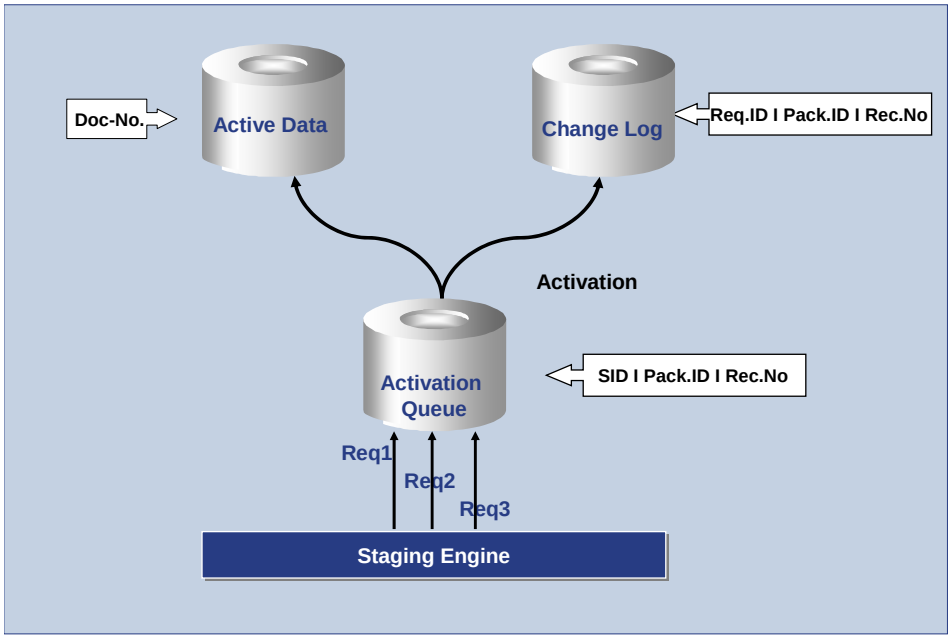
A standard DataStore object is represented on the database by three transparent tables:

Activation queue: Serves to save DataStore object data records that are to be updated, but that have not yet been activated. The data is deleted after the records have been activated.

Active data: A table containing the active data (A table).

Change log: Contains the change history for the delta update from the DataStore object into other data targets, such as DataStore objects or InfoCubes.

The tables of active data are built according to the DataStore object definition. This means that key fields and data fields are specified when the DataStore object is defined. The activation queue and the change log are almost identical in structure: the activation queue has an SID as its key, the package ID and the record number; the change log has the request ID as its key, the package ID, and the record number.



This graphic shows how the various tables of the DataStore object work together during the data load.

Data can be loaded performantly from several source systems simultaneously because a queuing mechanism enables a parallel INSERT. The key allows records to be labeled consistently in the activation queue.

The data arrives in the change log from the activation queue and is written to the table for active data upon activation. During activation, the requests are sorted according to their logical keys. This ensures that the data is updated to the table for active data in the correct request sequence.

See also: Example of Activating and Updating Data [External].

4.1.6.7.2.3 DataStore Objects for Direct Update

Definition

The DataStore object for direct update differs from the standard DataStore object in terms of how the data is processed. In a standard DataStore object, data is stored in different versions (active, delta, modified), whereas a DataStore object for direct update contains data in a single version. Therefore, data is stored in precisely the same form in which it was written to the DataStore object for direct update by the application. In the BI system, you can use a DataStore object for direct update as a data target for an analysis process. For more information, see Analysis Process Designer [External].

The DataStore object for direct update is also required by diverse applications, such as SAP Strategic Enterprise Management (SEM) for example, as well as other external applications.

Use

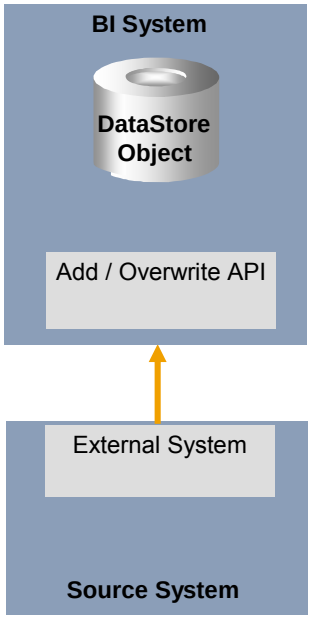
DataStore objects for direct update ensure that the data is available quickly. The data from this kind of DataStore object is accessed transactionally. The data is written to the DataStore object (possibly by several users at the same time) and reread as soon as possible.

It is not a replacement for the standard DataStore object. It is an additional function that can be used in special application contexts.

Structure

The DataStore object for direct update consists of a table for active data only. It retrieves its data from external systems via fill or delete APIs. See DataStore Data and External Applications [Page 163].

The load process is not supported by the BI system. The advantage to the way it is structured is that it is easy to access data. Data is made available for analysis and reporting immediately after it is loaded.



Creating DataStore Objects for Direct Update

When you create a DataStore object, you can change the DataStore object type under *Settings* in the context menu. The default setting is *Standard.* You can only switch between DataStore object types *standard* and *direct update* if data does not yet exist in the DataStore object.

Integration

Since you cannot use staging and fill DataStore objects for direct update with BI data (DataSources do not provide the data), DataStore objects are not displayed in the scheduler or in the monitor. However, you can update the data in DataStore objects for direct update to additional InfoProviders.

If you switch a standard DataStore object that already has update rules to *direct update*, the update rules are set to inactive and can no longer be processed.

Since no change log is generated, you cannot perform a delta update to the InfoProviders at the end of this process.

The DataStore object for direct update is available as an InfoProvider in the BEx Query Designer and can be used for analysis purposes.

4.1.6.7.2.4 Write-Optimized DataStore Objects

Definition

DataStore object that only consists of one table of active data. Data is loaded using the data transfer process.

Use

Data that is loaded into write-optimized DataStore objects is available immediately for further processing.

You use write-optimized DataStore objects in the following scenarios:

- You use a write-optimized DataStore object as a temporary storage area for large sets of data if you are executing complex transformations for this data before it is written to the DataStore object. Subsequently, the data can be updated to further (smaller) InfoProviders. You only have to create the complex transformations once for all data.
- You use write-optimized DataStore objects as the EDW layer for saving data. Business rules are only applied when the data is updated to additional InfoProviders.

The system does not generate SIDs for write-optimized DataStore objects and you do not need to activate them. This means that you can save and further process data quickly. Reporting is possible on the basis of these DataStore objects. However, we recommend that you use them as a consolidation layer, and update the data to additional InfoProviders, standard DataStore objects, or InfoCubes.

Structure

Since the write-optimized DataStore object only consists of the table of active data, you do not have to activate the data, as is necessary with the standard DataStore object. This means that you can process data more quickly.

The loaded data is not aggregated; the history of the data is retained. If two data records with the same logical key are extracted from the source, both records are saved in the DataStore object. The record mode responsible for aggregation remains, however, so that the aggregation of data can take place later in standard DataStore objects.

The system generates a unique technical key for the write-optimized DataStore object. The standard key fields are not necessary with this type of DataStore object. If standard key fields exist anyway, they are called *semantic keys* so that they can be distinguished from the technical keys. The technical key consists of the *Request GUID* field (0REQUEST), the *Data Package* field (0DATAPAKID) and the *Data Record Number* field (0RECORD). Only new data records are loaded to this key.

You can specify that you do not want to run a check to ensure that the data is unique. If you do not check the uniqueness of the data, the DataStore object table may contain several records with the same key. If you do **not** set this indicator, and you do check the uniqueness of the data, the system generates a unique index in the semantic key of the InfoObject. This index has the technical name "KEY". Since write-optimized DataStore objects do not have a change log, the system does not create delta (in the sense of a before image and an after image). When you update data into the connected InfoProviders, the system only updates the requests that have not yet been posted.

Use in BEx Queries

For performance reasons, SID values are not created for the characteristics that are loaded. The data is still available for BEx queries. However, in comparison to standard DataStore

objects, you can expect slightly worse performance because the SID values have to be created during reporting.

If you want to use write-optimized DataStore objects in BEx queries, we recommend that they have a semantic key and that you run a check to ensure that the data is unique. In this case, the write-optimized DataStore object behaves like a standard DataStore object. If the DataStore object does not have these properties, unexpected results may be produced when the data is aggregated in the query.

4.1.6.7.2.5 Creating DataStore Objects

Procedure

1. Select the InfoArea that you want to assign the DataStore object to, or create a new InfoArea. Choose *Modeling* → *Info Providers* → *Create InfoArea*.
2. In the context menu of the InfoArea, choose *Create DataStore Object*.
3. Specify a name and a description for the DataStore object, and choose *Create*.

   If you want to create a copy of an existing DataStore object, specify the DataStore object that you want to use as a template.

   You get to the DataStore object maintenance screen.
4. Copying InfoObjects:

   On the left side of the screen, there are various templates to choose from. These allow you to get a better overview in relation to a particular task. For performance reasons, the default setting is an empty template. You use the pushbuttons to select different objects as templates.

   On the right side of the screen you define the DataStore object. Using the drag and drop function, assign the InfoObjects in the key fields and in the data fields. You can select several InfoObjects at once. The system assigns navigation attributes automatically. These navigation attributes can be switched on to analyze data in BEx.

   Or:

   You are also able to insert InfoObjects without selecting a template in the left half of the screen. This is useful if you know exactly which InfoObjects you want to include in the DataStore object. To do this, choose *Insert InfoObjects* in the context menu for the node for key fields or data fields. In the dialog box that appears, you can enter and transfer up to 10 ten InfoObjects directly and or you can select them using input help. You can use drag and drop to reassign them.

   There has to be at least one key field and one data field.

   Additional restrictions:

   - You can create a maximum of 16 key fields (if you have more key fields, you can combine fields using a routine for a key field (concatenate)).
   - You can create a maximum of 749 fields
   - You can use 1962 bytes (minus 44 bytes for the change log)
   - You cannot include key figures as key fields

5. In the context menu of the *Data Fields* folder you can *Insert New Hierarchy Nodes*. In this way you can sort the data fields in a hierarchy. You then get a better overview of large quantities of data fields in query definition.
6. Using *Settings* you can make various settings and define the properties of the DataStore object. See also DataStore Object Settings [Page 156].
7. Under *Indexes*, call the context menu to create secondary indexes. This improves the load and query performance of the DataStore object. Primary indexes are created automatically by the system.

   If the values in the index fields uniquely identify each record in the table, select *Unique Index* from the dialog box. Note that errors can occur during activation if the values are not unique.

   The description of the indexes is specified by the system. To create a folder for the indexes, choose *Continue* from the dialog box. Now you can transfer the required key fields into the index folder using drag and drop.

   You can create a maximum of 16 secondary indexes. These are also transported automatically.

For more information see Indexes [External].

8. Use *Check* to make sure that the DataStore object is consistent.
9. Save the DataStore object, and activate it. When you activate the DataStore object, the system generates an export DataSource. You use this to update the DataStore object data to further InfoProviders.

4.1.6.7.2.5.1 DataStore Object Settings

Use

When creating and changing a DataStore object, you can make the following settings:

DataStore Object Type

Select the DataStore object type. You can choose between *standard*, *direct update* and *write-optimized*, where *standard* is the default value and *direct update* is only intended for special cases. You can switch the type as long as there is still no data in the DataStore object.

More information: Standard DataStore Objects [Page 151], DataStore Objects for Direct Update [Page 152], and Write-Optimized DataStore Objects [Page 154]

Check Table for InfoObjects

Here, the system displays the name of the InfoObject for which the DataStore object can also be the check table. You can specify this when creating an InfoObject. More information: Tab Page: Master Data/Texts [Page 80] in the InfoObject maintenance

Type-Specific Settings

The following settings are only available for certain DataStore object types:

For Write-Optimized DataStore Objects

Do Not Check Uniqueness of Data

This indicator is only relevant for write-optimized DataStore objects. With these objects, the technical key of the active tables always consists of the fields *Request*, *Data Package,* and *Data Record*. The InfoObjects that appear in the maintenance dialog in the *Semantic Key* folder form the semantic key of the write-optimized DataStore object.

If you set this indicator, **no** unique index is generated with the technical name "KEY" for the InfoObjects in the semantic key, and there can be multiple records with the same key in the active table of the DataStore object.

For Standard DataStore Objects:

Generating SID Values

With the *Generation of SID Values* indicator, you specify whether SIDs are created for the new characteristic values in the DataStore object when the data is activated. If you do not set the indicator, no SIDs are created and activation is completed faster.

Loading Unique Data Records

If you are only loading unique data records (data records with nonrecurring key combinations) into the DataStore object, the loading performance improves if you set the indicator *Unique Data Records* in the DataStore object maintenance.

The records are then updated more quickly because the system no longer needs to check whether the record already exists. You have to be sure that no duplicate records are loaded because this terminates the process.

If you set this indicator, the system automatically sets the indicator for creating SID values.

This DataStore object property is only supported if SID values are created during activation (see the *Creating SID Values* indicator). Otherwise, we recommend using a write-optimized DataStore object.

Automatic Further Processing

If you still use a 3.x InfoPackage to load data, you can activate several automatisms to further process the data in the DataStore object. If you use the data transfer process and process chains that SAP recommends you use, you cannot however use these automatisms.

We recommend that you always use process chains.

More information: Including DataStore Objects in Process Chains [External]

Settings for automatic further processing:

- Automatically Setting Quality Status to *OK*

  Using this indicator, you can specify that the system automatically sets the quality status of the data to OK after the data has been loaded into the DataStore object. Activate this function. You should only deselect this indicator if you want to check the data after it has been loaded.

- Activating the DataStore Object Data Automatically

  Using this indicator, you can specify that data that has the quality status *OK* is transferred from the activation queue into the table of active data, and that the change

log is updated. Activation is carried out by a new job that is started after data has been loaded into a DataStore object. If the activation process terminates, there can be no automatic update.

- Updating Data from DataStore Objects Automatically

  Using this indicator, you can specify that the DataStore object data is automatically updated. Once the data has been activated, it is updated to the connected InfoProviders. The first update is automatically an initial update. If the activation process terminates, there can be no automatic update. The update is carried out by a new job that is started once activation is complete.

  Only switch on automatic activation and automatic update if you are sure that these processes do not overlap.

For further settings, see Performance Optimization for DataStore Objects [Page 663].

4.1.6.7.2.6 Additional Functions in DataStore Object Maintenance

Documents

You can display, create or change documents for DataStore objects.

See also: Documents [External].

Version Comparison

You can compare changes in DataStore object maintenance for the following DataStore object versions:

- Active and modified version
- Active and Content version
- Modified and Content version

Transport Connection

You can select and transport DataStore objects. The system automatically collects all BI objects that are required to ensure a consistent status in the target system.

Where-Used List

You can determine which other objects in the BI system use a specific DataStore object. You can determine the effect of changing a DataStore object in a particular way and whether this is permitted at a given time.

BI Content

In BI Content DataStore objects, you can jump to the transaction for installing BI Content, copy the DataStore object, or compare it with the customer version. For more information, see BI Content (Versions) [External].

Structure-Specific Properties of InfoObjects

In the context menu of the InfoObject, you can assign specific properties to InfoObjects. These properties are only valid in the DataStore object you are currently processing.

The majority of these settings correspond to the settings that you can make globally for an InfoObject. For characteristics, these are *Display, Text Type* and *Filter Value Selection upon Query Execution.* See the corresponding sections under Tab Page: Business Explorer [Page 63].

You can also specify constants for characteristics.

By assigning a constant to a characteristic, you give it a fixed value. This means that the characteristic is available on the database (for validation, for example) but is no longer displayed in the query (no aggregation/drilldown is possible for this characteristic).

It is particularly useful to assign constants to compound characteristics.

Example 1:

The *storage location* characteristic is compounded with the *plant* characteristic. If only one plant is ever run within the application, you can assign a constant to the plant. The validation for the storage-location master table runs correctly using the constant value for the plant. In the query, however, the storage location only appears as a characteristic.

Example 2:

For an InfoProvider, you specify that only the constant 2005 appears for the year. In a query based on a MultiProvider that contains this InfoProvider, the InfoProvider is ignored if the selection is for year 2004. This improves query performance since the system knows that it does not have to search for records.

Special Case:

If the constant SPACE (type CHAR) or 00..0 (type NUMC) is assigned to the characteristic, specify character # in the first position.

For key figures, you have the settings *Decimal Places* and *Display*. See the corresponding sections under Tab Page: Additional Properties [Page 110].

Info Functions

Various information functions are available with reference to the status of the DataStore object:

- Log display for the save, activation, and deletion runs for the DataStore object
- DataStore object status in the ABAP/4 Dictionary and on the database
- Object catalog entry

Performance Settings:

You choose *Extras* → *DB Performance* to set the DB Memory Parameters [Page 659]. If you are using the database platform *DB2 UDB for UNIX, Windows and Linux*, you can also use Clustering [Page 662], for other databases use Partitioning [Page 660].

4.1.6.7.2.6.1 DB Storage Parameters

Use

You can maintain database storage parameters for PSA tables, master data tables, InfoCube fact- and dimension tables, and DataStore object tables.

Use this setting to determine how the system handles the table when it creates it in the database:

- Use *Data Type* to set in which physical database area (tablespace) the system is to create the table. Each data type (master data, transaction data, organization- and Customizing data, customer data) has its own physical database area, in which all tables assigned to this data type are stored. If selected correctly, your table is automatically assigned to the correct area when it is created in the database.

  We recommend you use separate tablespaces for very large tables.

  You can find information about creating a new data type in SAP Note 0046272 (Introduce new data type in technical settings).

- Via *Size Category*, you can set the amount of space the table is thought to need in the database. Five categories are available in the input help. You can also see here how many data records correspond to each individual category. When creating the table, the system reserves an initial storage space in the database. If the table later requires more storage space, it obtains it as set out in the size category. Correctly setting the size category prevents there being too many small extents (save areas) for a table. It also prevents the wastage of storage space when creating extents that are too large.

You can use the maintenance for storage parameters to better manage databases that support this concept.

You can find additional information about the data type and size category parameters in the ABAP Dictionary table documentation, under Technical Settings [External].

PSA Table

For PSA tables, you access the database storage parameter maintenance by choosing *Goto* → *Technical Attributes* in DataSource maintenance. In dataflow 3.x, you access this setting in transfer rule maintenance in the *Extras* menu.

You can also assign storage parameters for a PSA table already in the system. However, this has no effect on the existing table. If the system generates a new PSA version (a new PSA table) due to changes to the DataSource, this is created in the data area for the current storage parameters.

InfoObject Tables

For InfoObject tables, the maintenance of database storage parameters can be found in the InfoObject maintenance, in the *Extras* menu.

InfoCube/aggregate fact and dimension tables

For fact- and dimension tables, the maintenance of database storage parameters can be found in the InfoCube maintenance, in the *Extras* menu.

DataStore Object Tables (Activation Queue and Table for Active Data)

For DataStore object tables, the maintenance of database storage parameters can be found in DataStore object maintenance, in the *Extras* menu.

4.1.6.7.2.6.2 Multidimensional Clustering

Use

Multidimensional clustering (MDC) allows you to save the sorted data records in the active table of a DataStore object. Data records with the same key field values are saved in the same extents (related database storage unit). This prevents data records with the same key values from being spread over a large memory area and thereby reduces the number of extents to be read upon accessing tables. Multidimensional clustering greatly improves active table queries.

Prerequisites

Currently, the function is only supported by the database platform IBM DB2 Universal Database for UNIX and Windows.

Features

Multidimensional clustering organizes the data records of the active table of a DataStore object according to one or more fields of your choice. The selected fields are also indicated as MDC dimensions. Only data records with the same values in the MDC dimensions are saved in an extent. In the context of MDC, an extent is called a block.

The system creates block indexes from within the database for the selected fields. Block indexes link to extents instead of data record numbers and are therefore much smaller than row-based secondary indexes. They save memory space and can be searched through more quickly. This particularly improves performance of table queries that are restricted to these fields.

You can select the key fields of an active table of a DataStore object as an MDC dimension.

Multidimensional clustering was introduced in Release SAP NetWeaver 2004s and can be set up separately for each DataStore object.

For procedures, see Definition of Clustering [Page 161].

4.1.6.7.2.6.2.1 Definition of Clustering

Prerequisites

You can only change clustering if the DataStore object does not contain any data. You can change the clustering of DataStore objects that are already filled using the *Reclustering* function.

For more information, see Reclustering [Page 146].

Features

In the DataStore maintenance, select *Extras* → *DB Performance* → *Clustering*.

You can select MDC dimensions for the DataStore object on the *Multidimensional Clustering* screen.

Select one or more InfoObjects as MDC dimensions and assign them consecutive sequence numbers, beginning with 1. The sequence number shows whether a field has been selected as an MDC dimension and determines the order of the MDC dimensions in the combined block index.

In addition to block indexes for the different MDC dimensions within the database, the system creates the combined block index. The combined block index contains the fields of all the MDC dimensions. The order of the MDC dimensions can slightly affect the performance of table queries that are restricted to all MDC dimensions and those that are used to access the combined block index.

When selecting, proceed as follows:

- Select InfoObjects that you use to restrict your queries. For example, you can use a time characteristic as an MDC dimension to restrict your queries.
- Select InfoObjects with a low cardinality. For example, the time characteristic 0CALMONTH instead of 0CALDAY.

  You cannot select more than three InfoObjects.
- Assign sequence numbers using the following criteria:
  - Sort the InfoObjects according to how often they occur in queries (assign the lowest sequence number to the InfoObject that occurs most often in queries).
  - Sort the InfoObjects according to selectivity (assign the lowest sequence number to the InfoObject with the most different data records).

    Note: At least one block is created for each value combination in the MDC dimension. This memory area is reserved independently of the number of data records that have the same value combination in the MDC dimension. If there is not a sufficient number of data records with the same value combinations to completely fill a block, the free memory remains unused. This is so that data records with a different value combination in the MDC dimension cannot be written to the block.

    If for each combination that exists in the DataStore object, only a few data records exist in the selected MDC dimension, most blocks have unused free memory. This means that the active tables use an unnecessarily large amount of memory space. Performance of table queries also deteriorates, as many pages with not much information must be read.

Example

The size of a block depends on the PAGESIZE and the EXTENTSIZE of the tablespace. The standard PAGESIZE of the DataStore tablespace with the assigned data class DODS is 16K. Up to Release SAP BW 3.5, the default EXTENTSIZE value was 16. As of Release SAP NetWeaver 2004s the new default EXTENTSIZE value is 2.

With an EXTENTSIZE of 2 and a PAGESIZE of 16K the memory area is calculated as 2 x 16K = 32K, this is reserved for each block.

The width of a data record depends on the width and number of key fields and data fields in the DataStore object.

If, for example, a DataStore object has 10 key fields, each with 10 bytes, and 30 data fields with an average of 9 bytes each, a data record needs 10 x 10 bytes + 30 x 9 bytes = 370 bytes. In a 32K block, 32768 bytes/370 bytes could write 88 data records. At least 80 data records should exist for each value combination in the MDC dimensions. This allows optimal use of the memory space in the active table.

4.1.6.7.2.7 DataStore Data and External Applications

Use

Various functions allow you to read data from DataStore objects from external applications or to fill DataStore objects from external applications.

Features

The BAPI, BAPI_ODSO_READ_DATA_UC, for reading data, enables you to make DataStore data available to external systems. This BAPI can be used for all DataStore object types.

BAPI BAPI_ODSO_READ_DATA was used fort his purpose in the previous release. It is now obsolete.

The following APIs are available for direct update of DataStore objects (see DataStore Object for Direct Update [Page 152]):

RSDRI_ODSO_INSERT: Inserts new data (with keys not yet in the system).

RSDRI_ODSO_INSERT_RFC: see above, can be called up remotely

RSDRI_ODSO_MODIFY: inserts data having new keys; for data with keys already in the system, the data is changed.

RSDRI_ODSO_MODIFY_RFC: see above, can be called up remotely

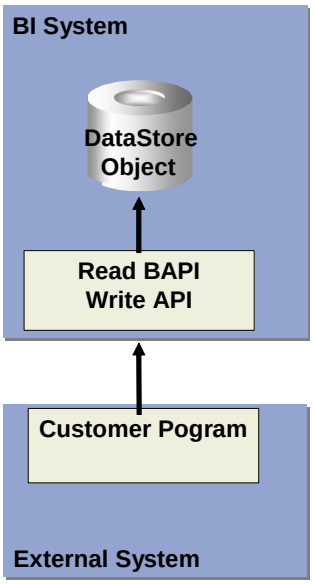
RSDRI_ODSO_UPDATE: changes data with keys in the system

RSDRI_ODSO_UPDATE_RFC: see above, can be called up remotely

RSDRI_ODSO_DELETE_RFC: deletes data

Also take a look at the extensive documentation for this function module in the BW system.

These APIs allow you to fill DataStore objects with data from external applications, for example, from Web applications.



You can also download from the DataStore object. You can display the DataStore table data from the DataStore object administration, and also download the data using the data browser so that you can use or display it in other applications. For more information see DataStore Object Content [External].

4.1.6.7.2.8 Questions and Answers

What are the benefits of loading requests in parallel?

Several requests can be updated more quickly in the DataStore object.

Can the processes for loading and activating requests be started independently of one another?

Yes. You can create a process chain that starts the activation process once the loading process is complete. More information: Including DataStore Objects in Process Chains [External]

Is there a maximum number of records that can be activated simultaneously?

No.

Can I change the loading method that is used to load the data into the DataStore object from a full update to a delta update?

No. Once a full update has been used to load data into the DataStore object, you are no longer able to change the loading method for this particular combination of DataSource and source system. One exception to this is updating a DataStore object to another (not yet filled) DataStore object if InfoProviders already exist that have been supplied with deltas from the DataStore object. You can run a full upload, which is handled like an initial, into the empty DataStore object and then load deltas on top of that.

Why is it that, after multiple data loads, the change log is larger than the table of active data?

The change log grows in proportion to the table of active data, because before and after-images of each new request are stored there. More information: Example for Activating and Updating Data [External] and the description of the delta process [Page 534].

Can I delete date from the change log once the data has been activated?

If a delta initialization is available for updates to connected InfoProviders, requests have to be updated before the corresponding data can be deleted from the change log. In the DataStore object administration, you can then call the Delete Change Log Data [External] function. You can schedule this process to run periodically.

However, you cannot immediately delete the data that you just activated, because the most recent deletion selection that you can specify is *Older Than 1 Day.*

Are locks set when I delete data from the DataStore object to prevent data being written simultaneously?

More information: Functional Constraints of Processes [External]

When is it useful to delete data from the DataStore object?

There are three options available for deleting data from the DataStore object: by request, selectively, and from the change log. To determine the best option, read the detailed description [External] of deleting data from DataStore objects.

When do I use the DataStore object for direct update?

You use this type of DataStore object to load data quickly without using the extraction and load processes in the BI system. More information: DataStore Objects for Direct Update [Page 152]

4.1.6.7.3 InfoObjects As InfoProvider

Definition

You can indicate an InfoObject of type characteristic as an InfoProvider if it has attributes and/or texts. The data is then loaded into the master data tables using the transformation rules.

You can only designate a characteristic as an InfoProvider if it contains texts or attributes. It is not possible to use transformation rules to load hierarchies.

Use

You can define transformation rules for the characteristic and use them to load attributes and texts.

You can also define queries for the characteristic (more exactly: for the master data of a characteristic) and then report using the master data.

In InfoObject maintenance, you can also select two-level navigation attributes (the navigation attributes for the navigation attributes of the characteristic) for this characteristic in the *Attribute* tab page. These are then available like normal characteristics in the query definition.

Integration

If you want to turn a characteristic into an InfoProvider, you have to assign an InfoArea to the characteristic. The characteristic is subsequently displayed in the InfoProvider tree in the Data Warehousing Workbench.

4.1.6.7.4 VirtualProviders

Definition

InfoProvider with transaction data that is not stored in the object itself, but which is read directly for analysis and reporting purposes. The relevant data can be from the BI system or from other SAP or non-SAP systems.

VirtualProviders only allow read access to data.

Use

Various VirtualProviders are available. You use these in different scenarios.

For more information, see:

- VirtualProvider Based on the Data Transfer Process [Page 166]
- VirtualProvider with BAPI [Page 170]

- VirtualProvider with Function Module [Page 171]
- Using InfoObjects as VirtualProviders [Page 175]

Note that the system does not run existing exits or customer and application extensions (customer exit, BTE, BAdI) for direct access to the source system.

4.1.6.7.4.1 VirtualProvider Based on the Data Transfer Process

Definition

VirtualProvider whose transaction data is read directly from an SAP system using a DataSource or an InfoProvider for analysis and reporting purposes.

Use

Use this VirtualProvider if:

- You require up-to-date data from an SAP source system.
- You only access a small amount of data from time to time.
- Only a few users execute queries simultaneously on the dataset.

Do **not** use this VirtualProvider if:

- You request a large amount of data in the first query navigation step, and no appropriate aggregates are available in the source system.
- Multiple users execute queries simultaneously.
- You frequently access the same data.

Structure

This type of VirtualProvider is defined based on a DataSource or an InfoProvider and copies its characteristics and key figures. Unlike other VirtualProviders, you do not need to program interfaces in the source system. To select data in the source system, you use the same extractors that you use to replicate data into the BI system. When you execute a query, every navigation step sends a request to the extractors in the assigned source systems. The selection of characteristics including the selection criteria for these characteristics is transformed, according to the transformation rules for the fields of the transfer structure. They are passed to the extractor in this form. The delivered data records pass through the transfer rules in the BI system and are filtered again in the query.

Since hierarchies are not read directly by the source system, they need to be available in the BI system before you execute a query. You can access attributes and texts directly.

Currently, the transformation only supports inverse transformations for direct assignment (without conversion routine) and the expert routine. Inverse transformations for other routine types and other rule types are also not yet implemented.

With more complex transformations such as routines or formulas, the selections cannot be transferred. It takes longer to read the data in the source system because the amount of data is not restricted. To prevent this, you can create an inversion routine for every transfer routine.

Inversion is not possible with formulas, which is why we recommend that you use routines instead of formulas.

Integration

To be assigned to this type of VirtualProvider, a source system must meet the following requirements:

- For a connection using a 3.x InfoSource, the BI Service API (included in Plug-In Basis) has to be installed. DataSources from the source system that are released for direct access are assigned to the InfoSource. There are active transfer rules for these combinations.
- The source system is Release 4.0B or higher.

See also:

Creating VirtualProviders Based on Data Transfer Processes [Page 167]

Creating VirtualProviders Based on 3.x InfoSources [Page 168]

4.1.6.7.4.1.1 Creating VirtualProviders Based on Data Transfer Processes

Prerequisites

If you are using a DataSource as the source for a VirtualProvider, you have to allow direct access to this DataSource.

Procedure

1. In the Data Warehousing Workbench under *Modeling,* choose the InfoProvider tree.
2. In the context menu, choose *Create VirtualProvider*.
3. As the type, select *VirtualProvider based on data transfer process for direct access.*

   In terms of compatibility, a VirtualProvider that is based on a data transfer process with direct access can also be connected to an SAP source system using a 3.x InfoSource. See Creating VirtualProvider Based on 3.x InfoSources [Page 168].

   The *Unique Source System Assignment* indicator controls whether this source system assignment needs to be unique. If the indicator is set, you can select a maximum of one source system in the assignment dialog. If the indicator is not set, you can select multiple source systems. In this case, the VirtualProvider acts like a MultiProvider.

   If the indicator is not set, characteristic 0LOGSYS is automatically added to the VirtualProvider when it is created. In the query, this characteristic allows you to select the source system dynamically: In each navigation step, the system only requests data from the assigned source systems whose logical system name fulfills the selection condition for characteristic 0LOGSYS.
4. Define the VirtualProvider by transferring the required InfoObjects. Activate the VirtualProvider.
5. In the context menu of the VirtualProvider, select *Create Transformation*. Define the transformation rules and activate them.
6. In the context menu of the VirtualProvider, select *Create Data Transfer Process*. *DTP for Direct Access* is the default value for the DTP type. Select the source for the

VirtualProvider. Activate the data transfer process. See Creating Data Transfer Process for Direct Access [Page 327].

7. Activate direct access. In the context menu of the VirtualProvider, select *Activate Direct Access*. In the dialog box that appears, choose one or more data transfer processes and select *Save Assignments*.

Result

The VirtualProvider can be used for analysis and reporting in the same way as any other InfoProvider.

4.1.6.7.4.1.2 Creating VirtualProviders Based on 3.x InfoSources

Use

For compatibility reasons, a VirtualProvider that is based on a data transfer process with direct access can also be connected to an SAP source system using a 3.x InfoSource.

Prerequisites

- Direct access must be allowed for the DataSource (in DataSource maintenance).
- This source system DataSource is assigned to the InfoSource.
- You have defined and activated the transfer rules for this combination. Note the special features with transfer routines:
  - You need to explicitly select the fields of the transfer structure that you want to use in the routine. See Creating Transfer Routines [Page 283].
  - If you have created a transfer routine, you can create an inversion routine for performance optimization.
  - If you use a formula, the selections in this field cannot be transferred. We recommend that you use a transfer routine instead.

Procedure

1. In the Data Warehousing Workbench under *Modeling,* choose the InfoProvider tree.
2. In the context menu, choose *Create VirtualProvider*.
3. As the type, choose *VirtualProvider based on the data transfer process* and enter your 3.x InfoSource.
4. You can set an indicator to specify whether a unique source system is assigned to the VirtualProvider. Otherwise you must select the source system in the query. In this case, characteristic (0LOGSYS) is added to the VirtualProvider definition.

   See also Characteristic Compounding with Source System ID [Page 87].
5. On the next screen you check the defined VirtualProvider and modify it, if necessary, before activation.
6. Activate direct access. From the context menu of the VirtualProvider, select *Activate Direct Access*.

7. Choose the *Source Systems* tab page for 3.x InfoSource. Select the source system and choose *Save Assignments.*

   The source system assignments are local to the system and are not transported.

Result

The VirtualProvider can be used in reporting in the same way as any other InfoProvider.

4.1.6.7.4.1.3 Processing Selection Conditions

In the context menu of the VirtualProvider, you choose the *Display Data* function. The system may display more data records than were actually selected. This depends on the processing type for the selection conditions on the characteristic. The *Display Data* function is only intended as a technical display; the system does not perform filtering. This can cause data records to appear in the display that are not selected by the selection conditions specified on the selection screen.

However, the correct result is displayed in the query. Surplus data records are filtered out again in the analytic engine after transformation.

Cause

For a VirtualProvider based on a data transfer process for direct access, there is always a transformation in the data flow between the source and the target (the VirtualProvider). Whether the selection conditions at the target can be passed back in full to the source using an inverse transformation depends on the complexity of the transformation. If it is not possible to pass back the exact selection conditions to the source, the selection conditions are simplified. This ensures that no data records in the source that correspond to the selection conditions after transformation are missed. In extreme cases this may mean that all data records are read from the source.

It may not be possible to pass back the selection conditions for a characteristic in the VirtualProvider to the source for the following reasons:

- The transformation consists of an expert routine or contains a start or end routine.
- The characteristic is filled using a rule of rule type *Formula*, *Routine* or *Read Master Data*.
- The characteristic is filled using a rule of rule type *Direct Assignment* and one of the following conditions applies:
  - The target characteristic and the source field are of type 'CHAR' but the target characteristic is longer that the source field.
  - The target characteristic has a conversion routine which is usually executed in the transformation.
  - The target characteristic uses one of the basic characteristics 0DATE, 0CALWEEK, 0CALMONTH, 0CALQUARTER or 0CALYEAR as a reference characteristic and the source field is either of type DATS or does not have the same type or length as the target characteristic.
  - The data type of the target characteristic is not the same as the data type of the source field (for example, NUMC characteristic filled from CHAR field) and the target characteristic does not use a basic characteristic 0DATE, 0CALWEEK, 0CALMONTH, 0CALQUARTER or 0CALYEAR as a reference characteristic.

- The characteristic is a time characteristic, is filled using a rule of rule type *Time Conversion* and at least one of the following is **not** true:
  - The time characteristic is an absolute time characteristic 0CALDAY, 0CALWEEK, 0CALMONTH, 0CALQUARTER or 0CALYEAR.
  - The source field is of type DATS or does not have the same type or length as the target characteristic.

4.1.6.7.4.2 VirtualProvider with BAPI

Definition

VirtualProviders whose transaction data is read for analysis and reporting from an external system using a BAPI.

Use

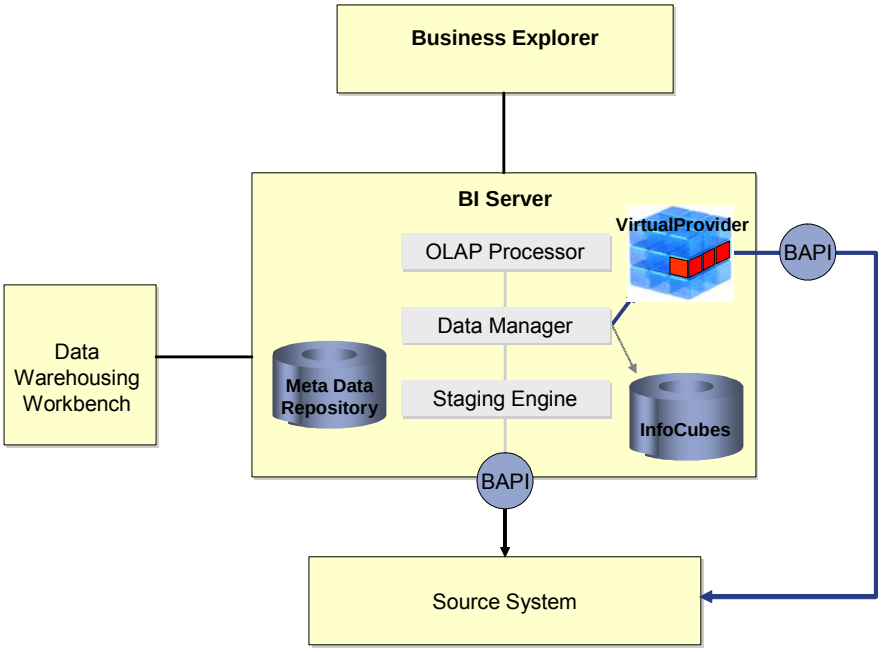
Using a VirtualProvider, you can carry out analyses on data in external systems without having to physically store transaction data in the BI system. You can, for example, use a VirtualProvider to include an external system from a market data provider.

When you start a query with a VirtualProvider, you trigger a data request with characteristic selections. The source structure is dynamic and is determined by the selections. The non-SAP system transfers the requested data to the OLAP processor using the BAPI.

This VirtualProvider allows you to connect non-SAP systems, in particular structures that are not relational (hierarchical databases). You can use any read tool that supports the interface for a non-SAP system.

Since the transaction data is not managed in the BI system, you have very little administrative effort on the BI side and can save memory space.

Structure



When you use a VirtualProvider to analyze data, the data manager calls the VirtualProvider BAPI, instead of an InfoProvider filled with data, and transfers the parameters.

- Selection
- Characteristics
- Key figures

The external system transfers the requested data to the OLAP processor.

Integration

To use a VirtualProvider with BAPI for analysis and reporting purposes, you have to perform the following steps:

1. In the BI system, create a source system for the external system that you want to use.
2. Define the required InfoObjects.
3. Load the master data.
4. Define the VirtualProvider.
5. Define the queries based on the VirtualProvider.

4.1.6.7.4.3 irtualProviders with Function Modules

Definition

A VirtualProvider with a user-defined function module that reads the data in the VirtualProvider for analysis and reporting purposes.

You have a number of options for defining the properties of the data source more precisely. According to these properties, the data manager provides various function module interfaces for converting the parameters and data. These interfaces have to be implemented outside the BI system.

Use

You use this VirtualProvider if you want to display data from non-BI data sources in BI without having to copy the dataset into the BI structures. The data can be local or remote. You can also use your own calculations to change the data before it is passed to the OLAP processor.

This function is used primarily in the SAP Strategic Enterprise Management (SEM) application.

In comparison to other VirtualProviders, this VirtualProvider is more generic. It offers more flexibility, but also requires a higher implementation effort.

Structure

You specify the type of the VirtualProvider when you create it. If you choose *Based on Function Module* as the type for your VirtualProvider, an extra *Detail* pushbutton appears on the interface. This pushbutton opens an additional dialog box, in which you define the services.

1. Enter the name of a function module that you want to use as the data source for the VirtualProvider. There are different default variants for the interface of this function module. One method for defining the correct variant, together with the description of the interfaces, is given at the end of this documentation.
2. You can choose options to support the selection conditions. You do this by selecting the *Convert Restrictions* option. These conversions only change the transfer table in the user-defined function module. The conversions do not change the result of the query because the restrictions that the function module does not process are checked later in the OLAP processor.

   Options:

   - No support: If this option is selected, no restrictions are passed to the function module.
   - Global selection conditions only: If this option is selected, only global restrictions (FEMS = 0) are passed to the function module. Other restrictions (FEMS > 0) that are created, for example, by setting restrictions on columns in queries, are deleted.
   - Hierarchies: If this option is switched on, the relevant InfoProvider supports hierarchy restrictions. This is only possible if the InfoProvider also supports SIDs.
   - Do not transform selection conditions: If this option is switched on, all selection conditions are passed to the function module, without being converted first.
3. Pack RFC: This option packs the parameter tables in BAPI format before the function module is called and unpacks the data table that is returned by the function module after the call is performed. Since this option is only useful with a remote function call, you have to define a logical system that is used to determine the target system for the remote function call, if you select this option.
4. SID support: If the data source of the function module can process SIDs, you should select this option.
   If this is not possible, the characteristic values are read from the data source and the data manager determines the SIDs dynamically. In this case, wherever possible,

restrictions that are applied to SID values are converted automatically into the corresponding restrictions for the characteristic values.

5. With navigation attributes: If this option is selected, navigation attributes and restrictions applied to navigation attributes are passed to the function module.
   If this option is not selected, the navigation attributes are read in the data manager once the user-defined function module has been executed. In this case, in the query, you need to have selected the characteristics that correspond to these attributes. Restrictions applied to the navigation attributes are not passed to the function module in this case.
6. Internal format (key figures): In SAP systems a separate format is often used to display currency key figures. The value in the internal format is different from the correct value in that the decimal places are shifted. You use the currency tables to determine the correct value for this internal representation.
   If this option is selected, the OLAP processor incorporates this conversion into the calculation.
7. Data quality settings
   - Data is delivered immediately: If you do not select the *Pack RFC* option, the function module interface contains parameter i_th_sfc with a numeric column ORDERBY or SORT. If this figure is not initial, it indicates the required sorting sequence of the field in the result. Choose the **Sorted data is delivered** option if the VirtualProvider delivers the data in the specified sequence.
   - "Exact" data is delivered: Choose the *"Exact" data is delivered* option, if the VirtualProvider always exactly observes all filters specified on the interface. In some cases, the VirtualProvider omits filters and returns a superset of the requested data.

Dependencies

If you use a remote function call, SID support has to be switched off and the hierarchy restrictions have to be expanded.

Different variants are allowed for the interface of the user-defined function module. These variants depend on the options you have chosen for the VirtualProvider:

- If *Pack RFC* is switched on, choose variant 1
- If *SID Support* is switched off, choose variant 2
- Otherwise, choose variant 3

Description of the Interfaces for User-Defined Function Modules

Variant 1:

```
IMPORTING
   infocube             LIKE bapi6200-infocube
   keydate              LIKE bapi6200-keydate
   return               LIKE bapiret2
TABLES
   selection            STRUCTURE bapi6200sl
   characteristics      STRUCTURE bapi6200fd
   keyfigures           STRUCTURE bapi6200fd
   data                 STRUCTURE bapi6100da
EXCEPTIONS
   communication_failure
   system_failure
```

This variant is the most general and the most straightforward. It is described in the documentation for function module BAPI_INFOCUBE_READ_REMOTE_DATA.

Variant 2:

```
IMPORTING
   i_infoprov         TYPE rsinfoprov
   i_keydate          TYPE rrsrdate
   i_th_sfc           TYPE rsdri_th_isfc
   i_th_sfk           TYPE rsdri_th_isfk
   i_t_range          TYPE rsdri_t_range
   i_tx_rangetab      TYPE rsdri_tx_rangetab
   i_first_call       TYPE rs_bool
   i_packagesize      TYPE i
EXPORTING
   e_t_data           TYPE standard table
   e_end_of_data      TYPE rs_bool
   e_t_msg            TYPE rs_t_msg
```

Variant 3:

SAP advises against using this interface.

The interface is intended for internal use only and is only mentioned here for completeness.

Note that the structures used in the interface may be changed by SAP.

```
IMPORTING
   i_infoprov        TYPE rsinfoprov
   i_keydate         TYPE rrsrdate
   i_th_sfc          TYPE rsdd_th_sfc
   i_th_sfk          TYPE rsdd_th_sfk
   i_tsx_seldr       TYPE rsdd_tsx_seldr
   i_first_call      TYPE rs_bool
   i_packagesize     TYPE i
EXPORTING
   e_t_data          TYPE standard table
   e_end_of_data     TYPE rs_bool
   e_t_msg           TYPE rs_t_msg
```

4.1.6.7.4.4 Using InfoObjects As VirtualProviders

Use

You can permit direct access to the source system for an InfoObject of type characteristic that you have selected for use as an InfoProvider.

This allows you to avoid loading master data. Note, however, that direct access to data has a negative impact on query performance. As with other VirtualProviders, you have to decide whether direct access to data is actually useful in the specific case in which you want to use it.

Procedure

1. You are in InfoObject maintenance. On tab page *Master Data/Texts,* assign an InfoArea to the characteristic and choose *Direct* as the type of master data access.
2. Activate the characteristic.
3. In the Data Warehousing Workbench under *Modeling,* choose the InfoProvider tree.
4. Navigate to your InfoArea. In the context menu of the attributes or texts for your characteristic, choose *Create Transformation*.
5. Define the transformation rules and activate them.
6. In the context menu of the attributes or texts for your characteristic, choose *Create Data Transfer Process*. *DTP for Direct Access* is the default value for the DTP type.
7. Select the source. Activate the data transfer process. See Creating Data Transfer Process for Direct Access [Page 327]. When you activate the DTP, the system automatically activates direct access.

Result

You can access data in the source system directly for this characteristic. Furthermore, you can create additional DTPs for the characteristic. If you create additional DTPs for the characteristic, you can deactivate direct access to a particular source system again,

depending on the source system from which you want to read data. In the context menu of the attributes or texts for your characteristic, choose *Activate Direct Access*.

4.1.6.7.5 InfoSets

Definition

Description of a specific kind of InfoProvider: InfoSet describes data sources that are defined as joins [External] of DataStore objects [Page 148], standard InfoCubes [Page 126] and/or InfoObjects [Page 57] (characteristics with master data). If one of the InfoObjects contained in the join is a time-dependent characteristic, the join is a time-dependent or temporal join.

An InfoSet is a semantic layer over the data sources.

Unlike the classic InfoSet, an InfoSet is a BI-specific view of data.

For more information, see the following documentation: InfoProviders [Page 125], Classic InfoSets [Page 209].

Use

With activated InfoSets you can define queries [External] in the BI suite.

InfoSets allow you to analyze the data in several InfoProviders by using combinations of master data-bearing characteristics, InfoCubes and DataStore objects. The system collects information from the tables of the relevant InfoProviders. When an InfoSet is made up of several characteristics you can map transitive attributes and analyze this master data.

You create an InfoSet using the characteristics Business Partner (0BPARTNER) – Vendor (0VENDOR) – Business Name (0DBBUSNAME) and can thereby analyze the master data.

You can use an InfoSet with a temporal join to map periods of time (see Temporal Joins [Page 199]). With all other types of BI object, the data is determined for the key date of the query but with a temporal join in an InfoSet, you can specify a particular point in time at which you want the data to be evaluated. The key date of the query is not taken into consideration in the InfoSet.

Structure

You can include any DataStore object, InfoCube or InfoObject of type *Characteristic with Master Data* in a join. A join can contain objects of the same object type, or objects of different object types. You can include individual objects in a join as many times as you want. Join conditions connect the objects in a join to one another (equal join-condition). A join condition determines the combination of individual object records that are included in the results set.

Integration

InfoSet Maintenance in the Data Warehousing Workbench

You create and edit InfoSets in the *InfoSet Builder*. See Creating InfoSets [Page 177] and Editing InfoSets [Page 179].

Queries Based on InfoSets

The BEx Query Designer supports a tabular (flat) display of queries. Use the *Table Display* pushbutton to activate this function.

In the BEx Query Designer, each InfoProvider in the join of type DataStore or characteristic bearing master data displays two separate dimensions (key and attribute). With InfoCubes, the dimensions of the InfoCube are mapped. These dimensions contain the fields and attributes for the selected InfoSet.

If the InfoProvider is an InfoObject of type *Characteristic*, all of the characteristics listed in attribute definition and all of the display attributes are assigned to the characteristics (and the compound characteristics, if applicable) in the *Key* dimension.

- The display attributes are only listed in the *Key* dimension.
- The independent characteristics are listed in both the *Key* and the *Attribute* dimensions.

If the InfoProvider is a DataStore object or an InfoCube, **no** field objects with the “exclusive attribute” property are listed in the directory tree of the InfoProvider.

If the join is a temporal join, there is also a separate *Valid Time Interval* dimension in the BEx Query Designer. See Temporal Joins [Page 199].

InfoSets offer you the *most recent reporting* for characteristics that bear master data; in reporting and analysis, the newest records are displayed, even if they are not activated yet. See Most Recent Reporting for InfoObjects [Page 205].

For more information about the technical details and examples of queries that use InfoSets, see Interpreting Queries Using InfoSets [Page 208] .

For more information about defining queries, see Query Design: BEx Query Designer [External]

Transport Connection

An InfoSet is connected to the BI transport system as a TLOGO object. For more information, see Transport Systems [External].

Definition and Delivery of Content

BI Content is defined and delivered in BI in the usual way. InfoSets are delivered in the D version and have to be activated by the customer (see BI Content (Versions) [External]).

4.1.6.7.5.1 Creating InfoSets

Prerequisites

Make sure that the objects for which you want to define the InfoSet are active. Create any required InfoObjects that do not exist already and activate them.

Instead of creating a new InfoSet, you can transfer one of the InfoSets that are delivered with SAP Business Content [Page 435].

Procedure

1. You are in the InfoProvider tree of the *Modeling* function area in the Data Warehousing Workbench.

   Choose the *Create InfoSet* function from the context menu of the InfoArea in which you want to create an InfoSet. The *Create InfoSet* dialog box appears.

2. Enter the following descriptions for the new InfoSet:
   - Technical name
   - Long name
   - Short name (optional)

3. In the *Start with InfoProvider* section, you determine which InfoProvider you want to use to start defining the InfoSet.
   - Select one of the object types that the system offers you:

     DataStore object

     InfoObject

     Standard InfoCube
   - Choose an object.

     If you want to choose an InfoObject, it must be a characteristic with master data. The system provides you with the corresponding input help.

4. Choose *Continue.* The first time you call the InfoSet Builder you can choose between two display modes: network (DataFlow Control) or tree (TreeControl). While the network display is clearer, the tree display can be read by the ScreenReader and is suitable for visually-impaired users. You can change this setting at any time using the menu path *Settings* → *Display*

   The *Change InfoSet* screen appears.

For more information, see Editing InfoSets [Page 179].

When you create an InfoSet, the system generates a corresponding entry for this InfoSet in the subtree of the InfoArea. The following functions are available from the context menu of this entry:

- Display
- Change
- Copy
- Delete
- Display data flow
- Object overview

If you want to create a new InfoSet you can also use transaction RSISET to call the *InfoSet Builder.* For more information, see Additional Functions in the InfoSet Builder [Page 183].

4.1.6.7.5.2 Editing InfoSets

Prerequisites

Before you can get to the screen where you edit InfoSets, the following prerequisites have to be met:

- You have created a new InfoSet.
- You have selected the *Change* function from the context menu of an InfoSet entry in the InfoProvider tree of the *Modeling* function area in the Data Warehousing Workbench.
- You have called the *InfoSet Builder* transaction, and selected the *Change* function. For more information see Additional Functions in the InfoSet Builder [Page 183].

Procedure

1. The *Change InfoSet* screen is displayed.

   Choose a layout for the InfoProvider tree:

   InfoAreas

   Related InfoProvider

   All DataStore Objects

   All InfoObjects

   All InfoCubes

   The default value is *Related InfoProviders*. Changed settings are personalized and stored if you leave InfoSet maintenance with F3. They are available in your next call.

   For more information on the screen layout, particularly the layout of the InfoProvider tree, see Screen Layout: Changing InfoSets [Page 184].

2. You use the *Where-Used List* function to determine which BI objects use the InfoSet that you have selected. The *Data Warehousing Workbench: Where-Used List* screen appears. This shows you the effects of changing the InfoSet. This helps you to decide whether you want to make these changes at this particular time.
3. You define or change the InfoSet by adding one or more InfoProviders to the join.

   In join control, there are several ways to add an InfoProvider:

   - From the InfoProvider tree:
     - Transfer the required InfoProvider by double-clicking on the appropriate entry in the InfoProvider tree.
     - Use drag and drop to transfer the required InfoProvider.
   - To add a particular InfoProvider irrespective of the current display of the InfoProvider tree, choose *Add InfoProvider*. The dialog box with the same name appears. Enter the required data.

     If you know the technical name of the InfoProvider that you want to add, this method is quicker than switching the layout of the InfoProvider tree.

When this function has been executed, the InfoProvider that you selected is displayed in the join control. For more information about the structure of the join control, see Join Control [Page 187].

4. Define the join conditions. For more information see Defining Join Conditions [Page 192].
5. You can get general information such as object version, date created and date changed by choosing *Goto* → *Global Settings*. You can also make various settings here. For more information, see:
   - Most Recent Reporting for InfoObjects [Page 205]
   - Left Outer Join [Page 194]
   - Local Aggregation [Page 206]
6. Click on the *Documents* pushbutton on the pushbutton toolbar to branch to the screen where you edit the documents for this InfoSet.
7. Use *Check* to check the correctness of the InfoSet definition. The log display is shown in the screen area under the join control.
8. Save the InfoSet. The log display is shown in the screen area under the join control.
9. Activate the InfoSet. When you activate the InfoSet, the system performs checks. The result of the activation is displayed in a log in the screen area under the join control.

Result

After you have activated the InfoSets, you can use them to define queries.

4.1.6.7.5.3 Special Features of InfoCubes in InfoSets

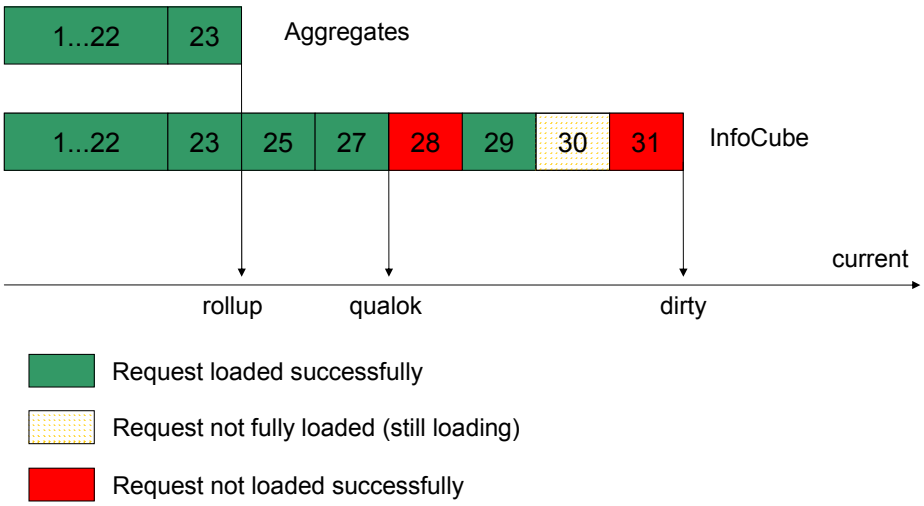
Use

InfoCubes are handled logically in InfoSets like DataStore objects. This is also true for time dependencies.

Features

Request Status

In an InfoCube, the system can read data with different request statuses. In the table view of the InfoCube, you make this setting in the context menu of the rows.



When you use InfoCubes in InfoSets, you can set the request up to which you want to rollup data in the aggregate (rollup), and the request up to which the data is qualitatively correct (qualok). You make these settings in InfoCube administration. The default for *qualok* is all green requests, where no yellow or red requests exist before them.

For example, requests 1-23 are rolled up into aggregates. Requests 1-27 are qualitatively correct.

In the context of the InfoCube in the InfoSet, the following alternatives are possible for the up-to-dateness of the data of an InfoCube:

- *Rolled Up Data* (rollup):

  The system only reads the rolled-up requests. This setting is the only setting that allows you to use aggregates under the conditions described in the following sections.

- *Up To Current Status* (qualok):

  In this case you cannot use aggregates, since the system also has to read data that has not been rolled up.

- *All Green Requests* (all):

  The system reads all correctly loaded requests. You cannot use aggregates.

- *All Requests* (dirty):

  The system reads all requests, including requests that were terminated and requests that were not loaded successfully, as well as requests that are currently being loaded. You cannot use aggregates.

  For large InfoCubes, only the *Rolled Up Data (Rollup)* option is useful. This is due to performance reasons. This is the default setting for each InfoCube in the InfoSet.

Using Aggregates

For queries based on an InfoSet with an InfoCube, the system decides at runtime whether aggregates can be used for the InfoCube. This is the case if all the required InfoObjects of the InfoCube exist in an aggregate. The following InfoObjects are required:

- The key figures of the InfoCube selected in the query
- The characteristics of the InfoCube selected in the query

- The characteristics required for a join with other InfoProviders in the InfoSet.

Furthermore, as a prerequisite for using aggregates, all the data required by an InfoCube must be readable using logical access. For an InfoCube within an InfoSet with InfoCubes, it is no longer possible to read part of the data from one aggregate and part of the data from another aggregate or the InfoCube itself.

The system cannot access BI accelerator indexes within an InfoSet.

Interpreting Record Counters for InfoSets with InfoCubes

The record counter of an individual InfoCube renders the number of records that physically exist and are affected by the selection. The record counter depends on the present aggregation status of the InfoCube. If you have chosen an aggregate, the record counter renders the selected number of records for the aggregate, not the number of records in the InfoCube from which the selected records in the aggregate were built.

For InfoSets with several InfoProviders, the key figure values are generally duplicated if all the characteristics affected by the join are not specified in the drilldown of the query. This also applies to the record counter.

Constraints

- For performance reasons, you cannot define an InfoCube as the right operand of a left outer join.
- SAP does not generally support more than two InfoCubes in an InfoSet. If you include more than two InfoCubes in an InfoSet, the system produces a warning. There are several reasons for this limitation:
  - Generally, the application server cannot create SQL statements with more than 64kb (in Unicode systems 32k characters). The more InfoCubes you use in an InfoSet, the quicker this limit is reached.
  - In contrast to the star schema (for which the potentially useful database access plans are limited by the table structure), several InfoCubes exist for a join, and several fact tables or DataStore object tables exist if you join InfoCubes with DataStore objects. There is no longer one large table at the center of the schema, and choosing a good access plan is much more difficult. Therefore, the average response time increases exponentially with the number of InfoCubes included.
  - If all of the characteristics affected by the join condition are not in the drilldown of a query, the key figure values of InfoCubes and DataStore objects are duplicated when you join them to InfoProviders (see SAP Note 592785). Therefore, interpreting the results of joins with non-unique InfoProviders becomes more difficult the more InfoProviders you include.

Design Recommendations

- To avoid problems caused by duplicated key figure values (see SAP Note 592785), we recommend that you only stage the key figures of one DataStore object or InfoCube of the InfoSet for the query (indicator in the first column in InfoSet maintenance).
- We recommend that you only use one InfoSet object (DataStore object, InfoCube, or master data table) with ambiguous characteristic values. This means that when you join a DataStore object with an InfoCube, as long as the InfoCube contains the visible key figures, all the key characteristics of the DataStore object are used in the join condition for the InfoCube. Equally, when joining a master data table with compounding to an

InfoCube, all of the key characteristics of the master data table are joined with the characteristics of the InfoCube.

4.1.6.7.5.4 Additional Functions in the InfoSet Builder

You can also use transaction RSISET to call up the InfoSet Builder when you want to edit an InfoSet. Select the InfoSet that you want to edit. Input help is available for this. Additional functions are also available to help you edit and manage your InfoSet.

Compare

You use this function from the main menu to check if the InfoProviders used in the InfoSet have been changed and the InfoSet needs to be adjusted as a result. For more information, see Comparing InfoSets [Page 186].

Jump to object maintenance

You use the *InfoObjects,* *DataStore Objects* and *Standard InfoCube* functions to jump to the maintenance screen for the InfoProviders included in the InfoSet definition.

Info functions

There are various info functions on the status of the InfoSets:

- The *Object Catalog Entry*
- The log display for the save, activate, and delete runs of the InfoSet.

Tree Display

With this function you can display in a tree structure all the properties of the A, M and D versions, if they exist, for the selected InfoSet:

- *Header data*
- InfoProvider and its fields
- O*n* condition
- *Where* condition

The display is empty, if no active version is available.

Version comparison

You use this function to compare the following InfoSet versions:

- The active (A version) and modified (M version) versions of an InfoSet
- The active (A version) and content (D version) versions of an InfoSet
- The modified (M version) and content (D version) versions of an InfoSet

The *Display InfoSet* screen appears. Depending on which option you choose, the system displays either all of the differences between the two versions of the selected InfoSet or all of the properties of both versions in a tree structure.

Transport connection

You use this function to transport an InfoSet into another system.

The *Data Warehousing Workbench: Transport Connection* screen appears.

The system has already collected all the BI objects that are needed to guarantee the consistency of the target system.

InfoSet data display

You use this function to access the data target browser. If you have already loaded data into the InfoProviders included in the InfoSet, you can display this data.

Delete

You use this function to delete an existing InfoSet.

Copy

You use this function to copy an existing InfoSet and, if necessary, edit it further.

4.1.6.7.5.5 Screen Layout: Changing InfoSets

The *Change InfoSet* screen has the following areas:

InfoProvider Tree

On the left-hand side of the screen, all of the InfoProviders that you are able to use in the InfoSet definition are displayed.

The options in the following table control how the InfoProvider tree is displayed:

InfoProvider Tree	What You Need to Know
InfoArea	This tree contains all the DataStore objects, InfoCubes and InfoObjects that are characteristics (with master data), that are available in the BI system. These are arranged according to InfoArea. Choose the *Expand Nodes* option from the context menu of an InfoArea to display all of the objects that belong to this particular InfoArea, in a hierarchy . If you do not want to display the lower levels, choose the *Collapse Nodes* option from the context menu of the InfoArea.
InfoProviders Used	This tree contains those InfoProviders that can be included in the join and for which you can define a join condition for a join that already exists in the InfoProvider. • InfoObjects that are characteristics (with master data) and that are either already included as InfoProviders in the join, or are attributes of an InfoProvider in the join.

	• DataStore objects whose keys contain an InfoObject that is either already included as an InfoProvider in the join, or that is an attribute of an InfoProvider in the join. • InfoCubes that are either already included in the join, or have InfoObjects in their dimensions that are already in the InfoSet. The *Related InfoProviders* tree contains the following objects in particular: • InfoProviders that are already available in the join, because each InfoProvider can be included in the join more than once. • InfoProviders for which you can define a join condition with the first InfoProvider that you choose when the InfoSet is created.
All DataStore Objects	This tree contains all the DataStore objects that are available in the BI system.
All InfoObjects	This tree contains all the InfoObjects that are characteristics (with master data) that are available in the BI system.
All InfoCubes	This tree contains all the standard InfoCubes that are available in the BI system.

The default layout is the *Related InfoProviders* tree. SAP recommends that you use the default layout.

The system is, however, able to store one of the alternative layouts as a personal setting. You must exit the InfoSet maintenance using F3.

Although, in principle, you can include every DataStore object, every InfoCube and every InfoObject that is a characteristic (with master data) in a join, you must remember that not every DataStore object or characteristic supports the definition of a join condition and that this is a prerequisite for activating the InfoSet.

You access the InfoProvider maintenance screen for a particular InfoProvider by clicking on the corresponding option in the context menu.

To access the maintenance screen for a DataStore object, call the context menu for the respective InfoProvider and choose the *Display DataStore Object* option.

Join Control

On the right-hand side of the screen there is a join-control. You use the join-control to display the InfoProviders used and the relationships between them. For more information, see Join-Control [Page 187].

Area for Logs and Text Maintenance

In the area underneath the join control, logs and texts that you want to maintain are displayed as and when you require them.

4.1.6.7.5.6 Matching InfoSets

Use

If changes have been made to InfoProviders that are used in InfoSets, you must compare the InfoSet with the modified InfoProviders and make any changes necessary to make the two match again. When you call the InfoSet Builder, the system checks whether the InfoProviders that are used have been changed at all. At this point, you have the option of comparing and adjusting the InfoSets as required. This needs to be done, because otherwise there is a risk that the data will be inconsistent.

Features

The match (compare and adjust) function for InfoSets checks the following:

In the case of DataStore objects and master data-bearing characteristics

- Whether attributes/data fields have been added or removed.

In the case of DataStore objects in particular -

- Whether changes have been made to the key (whether key fields have been added or removed)

In the case of master data-bearing characteristics in particular -

- Whether changes have been made to the compounding (whether attributes have been added or removed)
- Whether changes have been made to the time-dependency of attributes
    - If a new time-dependent attribute or an existing time-independent attribute that has been converted into a time-dependent attribute is added to a characteristic that until now has contained only time-independent attributes, meaning that until now it has been a time-independent characteristic. In the InfoSet it must be made clear that this characteristic is now time-dependent.
    - If the time-dependent attributes belonging to a characteristic are all converted into time-independent attributes, or all the time-dependent attributes are removed from the InfoProvider. In the InfoSet it must be made clear that this characteristic is now time-independent.

In the case of InfoCubes in particular

- Whether new dimensions have been added to the InfoCube
- Whether InfoCube dimensions have been deleted.

Explanation of the Log

Green means that you do not need to compare and adjust (match) the objects.

Yellow means that the objects do need to be compared and adjusted (matched) and that this process can be carried out automatically. Choose *Match*.

Red means that the objects need to be compared and adjusted (matched) but that this process cannot be carried out automatically. You have to change and reactivate the InfoSet manually in the InfoSet Builder.

Dependencies

In the following situations, when the traffic light shows red, you need to make changes to the InfoSet definition manually:

- If attributes that have been removed from their corresponding InfoProvider are still joined by a join condition to other objects or attributes, the system is not able to compare and adjust (match) the objects automatically until you have removed this link in the InfoSet Builder.
- If a temporal operand has been set for attributes that have been removed from their corresponding InfoProvider, you first have to reset this indicator in the InfoSet Builder.

When you have made these changes to the InfoSet, restart the match (compare and adjust) process.

Activities

You use transaction RSISET to call the match (compare and adjust) function. In the main menu, choose *InfoSet → Match.* The system checks the InfoProviders that are used in the InfoSet and produces a log giving details of the results of the check. You can decide whether a match should be performed.

4.1.6.7.5.7 Join Control

Definition

An area of the screen belonging to the *InfoSet Builder*. The InfoProviders that are included in the join are displayed in the join control.

Use

You define join conditions in the join control. There must be valid join conditions before the system is able to activate the InfoSet. For more information, see Defining Join Conditions [Page 192].

The first time you call the InfoSet Builder you can choose between two display modes:

network (DataFlowControl) or tree (TreeControl). While the network display is clearer, the tree display can be read by the ScreenReader and is more suitable for visually-impaired users. You can change this setting at any time by choosing *Settings → Display.* Changes take effect the next time you call the InfoSet Builder.

To edit two InfoProviders from one InfoSet, you can call a separate join control. For more information, see Editing InfoProviders in the Join Control [Page 198].

Structure

The same functions are available in both display modes. However, the network display is more commonly used, as it gives a clearer overview. This is why the differences between the tree display and the network display are only briefly addressed here. After this section, only the network display will be described.

Special Features of the Tree Display

The InfoProvider is displayed in a tree structure in the join control. The symbol *Time-Dependency Deactivated* indicates the option of a time dependency. An existing left outer join is indicated using the flag. You can display a join in the right-hand side of the screen by double-clicking on an InfoObject. You can set the join condition with the *Selection* indicator.

Displaying an InfoProvider in the Join Control

InfoProviders are displayed as a **table** in the join control. A symbol in the header indicates that an InfoProvider is time-dependent. The inactive version of this symbol indicates the option of a time dependency.

Depending on the type of InfoProvider, the following information is displayed in the **rows** of the table:

- for DataStore objects and InfoCubes: Each with a field (key or data field), for InfoCubes there are also some dimension rows
- for InfoObjects: The InfoObject itself, compounded characteristics, or an attribute

  Since InfoObjects are used to define the fields for DataStore objects, InfoCubes and the attributes of InfoObjects, each row ends with an InfoObject, except for InfoCubes that also have dimension rows.

InfoObjects are described as follows in the **columns** of the table:

Column	What You Need to Know
Use field	Field selection for an InfoSet: If there is an indicator in this checkbox, the indicated field or attribute of an InfoProvider is released for use in reporting. This means that it is available in the BEx Query Designer to be used for defining queries. The indicator is set by default. You can restrict the number of available fields or attributes from an InfoProvider by removing this indicator. If an InfoObject has the property “exclusive attribute”, the checkbox for selecting this field object in the join control is not ready for input. This is because the respective characteristic can only be used as a display attribute for another characteristic. This restriction does not apply to indicators. In the BEx Query Designer these display attributes are **not** available for the query definition in the InfoProvider directory tree (see Defining a New Query [External]). In order to add these field objects in queries, you must deactivate the property *Attribute Only* in the InfoObject maintenance. (See Tab: General [Page 61]) This may influence the performance of database access.
Key field, additional field, dimension	The symbol means

	• A key field for DataStore objects • For InfoObjects, the InfoObject itself or a compounded characteristic The symbol means additional attributes, for time-dependent InfoObjects • For the start of a valid time interval (*valid from*) • For the end of a valid time interval (*valid to*) • And for all InfoProviders • Key dates The symbol means • For InfoCubes: a dimension
Technical name	
Object type (represented by the corresponding symbol)	Examples: Characteristic Key Figure Unit Time Characteristic
Description	Long text description
Key date	This column is only filled for D type (date) fields or attributes of an InfoProvider, and for time characteristics, from which a key date is derived (0CALWEEK, 0CALMONTH, 0CALQUARTER, 0CALYEAR, 0FISCPER, 0FISCYEAR). If the indicator is set in this checkbox, the InfoObject is used as a temporal operand. The indicator is set to empty by default. If it is set and a key date can be derived, the additional fields mentioned above are added to the InfoProvider. See Temporal Joins [Page 199].

The following functions are available from the **context menu** of a table entry:

- *Define Time-dependency*

  This enables you to define valid time intervals. The appropriate characteristics are offered to you using input help. For more information, see Temporal Joins [Page 199].

- *Request status*

  This function is only available for InfoCubes. For more information, see Special Features of InfoCubes in InfoSets [Page 180].

- *Delete Object*

  Choose this function to delete an object from the join control.

- *Left Outer Join* or *Inner Join*

  For more information on the left outer join operator, see Defining Join Conditions [Page 192].

- *Select All Fields*

  If you choose this option, all fields or attributes of an InfoProvider are released for reporting. The indicators are respectively set in the column *Use Field*.

- *Deselect All Fields*

  If you choose this option, all indicators are removed from the column *Use Field*.

Displaying Join Conditions in the Join Control

A join condition is displayed as a line that connects exactly one InfoObject within a row from one object, with exactly one InfoObject within a row from another object.

For more information, see Defining Join Conditions [Page 192].

Navigating in the Join Control

Location of the individual objects

The system inserts each object from a fixed, predetermined default size into the join control.

If you want to insert a new object next to a specific table, select the table you want. The system inserts the new object at the same level, to the right of the selected table.

If no table is selected, the system inserts the new object at the same level, to the right of the table furthest away on the right.

You are able to position each DataStore object and each InfoObject freely in the join control. Position the cursor over the header of the object, press the left mouse-button, and keeping the button pressed down, drag the object to its new position.

The positioning of the individual objects within the join control does **not** affect the processing of the join.

Size of the individual objects

Each time you click on the *Zoom in* icon, the view is enlarged by 10%.

Each time you click on the *Zoom out* icon, the view is reduced by 10%.

The *Auto-Arrange* function automatically arranges the objects into an overview.

Navigator

You click on the *Hide/Display Navigator* function to access the navigation help.

This function is also available from the context menu of the join control.

The navigator is particularly useful if not all the objects are visible at the same time.

- If you want to change the section of the screen that is displayed, you move the red frame in the navigator.
- If you want to change the size of the objects, you adjust the dimensions of the frame itself:

  Reducing the frame has the same effect as the zoom-in function.

Enlarging the frame has the same effect as the zoom-out function.

You can also choose the functions *Zoom in, Zoom out* and *Show/Hide Navigator* in the context menu of the join control.

Changing the Descriptions

The descriptive texts that are used in the metadata repository for the InfoProviders and their attributes are also used in the join control.

If you use InfoProviders or InfoObjects more than once as attributes in the join, it helps if you change the descriptive texts for the purposes of the InfoSet. This enables you to identify the individual objects more easily.

Choose the *Change Description* function. An overview of all the texts is displayed beneath the join control. You are able to change each of these texts.

The following functions are available:

Function	What You Need to Know
All Objects	A selection of the texts for • a single InfoProvider in the join • all the objects in the join
Transfer	Transfers the texts in the display to the join control.
Get All Original Texts	Undoes the changes made to the texts. If you click on the *Transfer* function at this stage, the system re-inserts the descriptions from the metadata repository.

Delete

Select one or more objects that you want to delete from the join and click on the *Delete* function.

Saving a Join as a .jpg File

Choose the *Save as jpg* function to save your join definition as a graphic, in the jpeg file format, on a PC.

Print

Choose the *Print* function to print a copy of your join definition.

Show/Hide Technical Names

You can use this function to show alias names for fields and tables / InfoProviders. These alias names are necessary in InfoSets, for example to be able to map self joins. Field alias names start with *F* and are followed by a five digit number starting with *1*. The names are numbered sequentially. Table aliases start with *T* followed by a number starting with *1*. These are also numbered sequentially. In both cases, the maximum number possible is 99999.

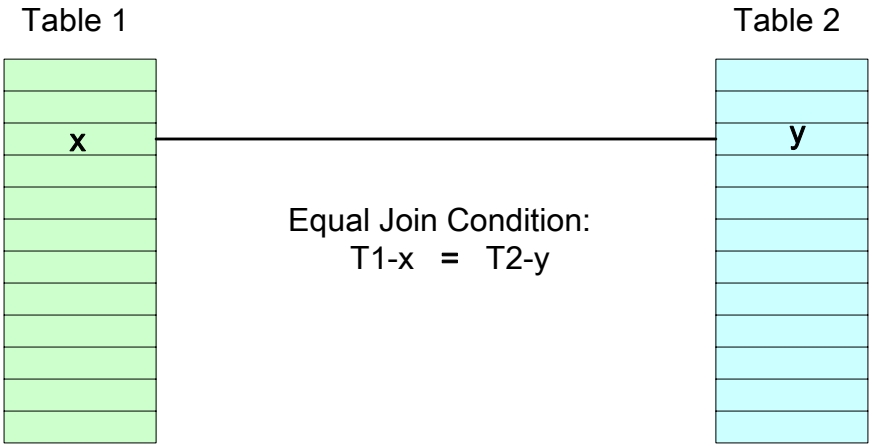
4.1.6.7.5.8 Defining Join Conditions

Use

A join condition determines the combination of individual object records that are included in the results set.

Before an InfoSet can be activated, the join conditions have to be defined in such a way (as equal join condition) that all the available objects are connected to one another either directly or indirectly.

An equal join condition is only met when the value is the same as the value of the associated error.



Technical requirement of a join condition is that the data type and length of the two joined InfoObjects are the same.

Usually, however, only rows containing a common InfoObject, or rows containing InfoObjects that share the same basic characteristic, are connected to one another.

Connect tables T1 and T2 using a join and set as a join condition that the F1 field from T1 must have the same value as F2 from T2. For a record from table T1, the system determines all records from T2 for which F2(T2) = F1(T1) is true. In principle, as many records from T2 can be found as required. If one or more records are found, the corresponding number of records is included in the results set, whereby the fields from T1 contain the values from that particular record in T1, and the fields from T2 contain the values of the records found in T2.

Procedure

There are various ways to define join conditions:

Using a Mouse Click

Position the cursor over a row in an InfoObject. Press the left mouse button and, keeping the left mouse button pressed down, trace a line between this row and a row in another object. Providing that the join condition between the two rows that you have indicated is valid, the system confirms the join condition by displaying a connecting line between the two rows.

Using the *Link Maintenance* Pushbutton

The *Link Maintenance* dialog box appears.

In a tree structure on the left-hand side of the screen, all of the InfoProviders that are already included in the join are displayed along with their fields or attributes. If you double-click on one of these fields or attributes, the system displays on the right-hand side of the screen all of the fields or attributes with which you are able to create a join condition.

In the *Selection* column, set one or more of the indicators for the fields or attributes for which you want to create a join condition. The system generates valid join conditions between the fields or attributes that you specify.

You use the *Delete Links* pushbutton to undo all of the join conditions.

You use the *All Characteristics* or the *Basic Characteristics Only* pushbutton to choose between the corresponding display variants.

We recommend that you use the *Basic Characteristics Only* option. The *All Characteristics* setting displays all of the technical options involved in a join. If you are unable to find a join condition on the basic characteristic level, then the *All Characteristics* setting is useful, but this is an exceptional case.

When you have finished making your settings, click on the *Continue* icon.

We recommend this method because the system searches for all the possible join conditions for any field or attribute that the user specifies, ensuring that the join conditions are defined without errors.

Using the Context Menu *Left Outer Join – Inner Join*

Usually inner join operators connect all the objects in a join to one another. You can also connect objects with any of the many other objects by using a left outer join operator. This function is not available for InfoCubes.

For more information about usage and special features, see Left Outer Join [Page 194].

If you want to use a left outer join operator to connect an object, select the object and choose the corresponding function from the context menu.

The system displays all of the valid join conditions that originate from this object. The connecting lines that represent these join conditions are labeled as *Left Outer Join* . InfoProviders that are connected using a left outer join condition are differentiated by color from those that are connected using an inner join operator.

If you use a left outer join operator to connect two objects, you have to make sure that all join conditions are linked except for these two objects with the formulation of join conditions.

Note: You cannot add an object that you have already connected by using the left outer join operator, to another object.

If you want to use an inner join operator instead of a left outer join operator, select the object that you want to connect, and choose the *Inner Join* option from the context menu. The system displays all the valid join conditions that originate from this object, using unlabeled connecting lines.

Result

Once the join conditions have been defined so that they connect all the available objects to one another, either directly or indirectly, the InfoSet can be saved as active.

Choose *Check* in the pushbutton toolbar to find out if the preconditions for the activation have been met.

For objects that are joined by a left outer join operator, there is one extra condition that has to be met; all the other objects must be connected to one another either directly or indirectly.

4.1.6.7.5.8.1 Left Outer Join

Use

When defining InfoSets, the objects are usually linked using inner join operators. However, you can also use left outer joins.

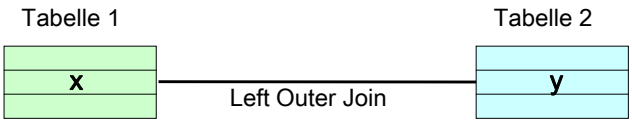
Left outer joins are not possible for InfoCubes. This would have an adverse affect on performance.

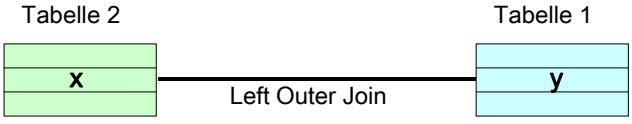
Inner join and left outer join are only different where one of the involved tables does not contain any suitable record that meets the join conditions.

With an **inner join** (table1 *inner join* table2), no record is included in the results set in this case. However, this means that the corresponding record from table 1 is not considered in the results set.

In this case, with a **left outer join** (table1 *left outer join* table2), exactly one record is included in the results set. In this record, the fields from table1 contain the values of the record from table1, and the fields from table2 are all filled with the initial value.

The order of the operands is very important for a left outer join. This means that the following joins describe different results sets:





The sequence must be adhered to when defining a left outer join. For an inner join, the sequence of the operands is not important.

You should always use a left outer join when:

1. It cannot be ensured that at least one suitable record is found in the involved table in accordance with the join conditions, and
2. You want to avoid records being included in the results set, since one of the tables returns no entry.

As a result of the above points it is possible to assume that a left outer join would be the best option, since it has many advantages. However, it should be made clear that a left outer join is only to be used when it is really necessary. This is because it has a significantly negative affect on performance in comparison to an inner join and is thus subject to certain restrictions (see features).

Features

If a left outer join is used, the following restriction applies to the right table (right operand):

- Only join conditions with exactly one other table can be defined and
- This table in turn cannot be a right table (right operand) of a left outer join.

Tables connected with left outer joins always form the end of a chain of tables. In this way, as many tables as you want can be linked in an InfoSet with a left outer join to a core of tables that are connected using inner joins.

The restrictions on the definition of left outer joins are due to the technical limitations of databases. These restrictions do not apply to inner joins.

Include Filter Value in Condition

In the global properties of the InfoSet, you can use an indicator to determine how a condition on a field of the left outer table is implemented in the SQL statement.

This affects the query results:

- After you have set the indicator *left outer: include filter value in the on-condition*, the condition/restriction is included in the on-condition in the SQL statement. The condition is then evaluated before the join.
- If you do not set the indicator, the condition/restriction is included in the where-condition. The condition is then only evaluated after the join.

The indicator is set to empty by default. To see the affects that this indicator has on the result, see Examples of Condition Conversion [Page 196].

Example

A typical example would be a DataStore object that contains a characteristic, for example PLANT, alongside key figures in its data part. In an InfoSet, a join between this DataStore object and the characteristic PLANT is defined so that the system can access the attributes of PLANT in reporting. A query based on this DataStore object evaluates the key figures existing in the DataStore object.

If an inner join is now used and if a DataStore object record contains a value for PLANT for which there is no entry in the corresponding master data table, this record is not included in the results set. Correspondingly, the key figures of this record would not be considered. If, on the other hand, a left outer join (DataStore object *left outer join* PLANT) is used, the corresponding record is considered. However, in this case, all attributes of the (non-existent) characteristic PLANT are initial. The correct behavior depends on the type of evaluation required. Both cases are valid.

The table used for selecting (the main table) may never be indicated as the left outer join

See also:

Defining Join Conditions [Page 192]

4.1.6.7.5.8.2 Examples of Condition Conversion

The following examples show how the *Left Outer: Include Filter Value in On-Condition* indicator affects the query results.

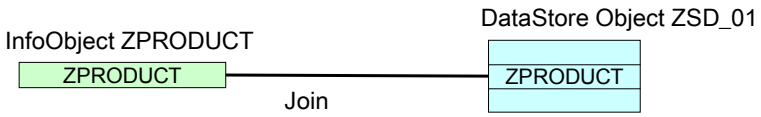
Characteristic ZPRODUCT (T00001), which has master data, contains two data records.

#Field (ZPRODUCT)
A
B

DataStore object ZSD_01 (T00002) contains three data records:

#Field (ZPRODUCT)	#Date (0DATE)	ABC Indicator (ABCKEY)
A	27.09.2003	X
A	01.04.2003	X
C	17.05.2003	X

In the InfoSet, the two InfoProviders are joined: ZPRODUCT- ZPRODUCT with ZSD_01-ZPRODUCT



Note the following cases:

Case 1

The objects are joined using an inner join. All the fields are output in the query and a restriction is applied to the date (01.04.2003). If all the objects in the InfoSet are joined using an inner join, this indicator does not effect the SQL statement that is generated, or the end result. It does not matter whether the condition is executed before the join is evaluated or afterwards. The result is the same, whether the restrictions apply in the on-condition or the where-condition.

In both cases, the result is:

#Field (ZPRODUCT)	#Field (ZPRODUCT)	#Date (0DATE)	ABC Indicator (ABCKEY)
A	A	01.04.2003	X

Case 2

The objects are joined using a left outer join (the outer condition is set for the DataStore object). All the fields are output in the query and a restriction is applied to the date (01.04.2003).

In this case, we assume that the indicator is initial. The restriction is included in the where-condition and is evaluated after the join.

This means:

First, the join is built. The results are as follows:

#Field (ZPRODUCT)	#Field (ZPRODUCT)	#Date (0DATE)	ABC Indicator (ABCKEY)
A	A	27.09.2003	X
A	A	01.04.2003	X
B			

The restriction is applied to these results (date = 01.04.2003).

The results are as follows:

#Field (ZPRODUCT)	#Field (ZPRODUCT)	#Date (0DATE)	ABC Indicator (ABCKEY)
A	A	01.04.2003	X

Case 3

The objects are joined using a left outer join (the outer condition is set for the DataStore object). All the fields are output in the query and a restriction is applied to the date (01.04.2003).

In this case, we assume that the indicator is not initial. The restriction is included in the on-condition and is evaluated before the join.

This means:

First, the restriction is applied. The following record is produced for the DataStore object:

#Field (ZPRODUCT)	#Date (0DATE)	ABC Indicator (ABCKEY)
A	01.04.2003	X

In the second step, the join is performed. The results are as follows:

#Field (ZPRODUCT)	#Field (ZPRODUCT)	#Date (0DATE)	ABC Indicator (ABCKEY)
A	A	01.04.2003	X
B			

The restriction is applied in the on-condition. The result is as follows:

#Field (ZPRODUCT)	#Field (ZPRODUCT)	#Date (0DATE)	ABC Indicator (ABCKEY)
A	A	01.04.2003	X
B			

4.1.6.7.5.9 Editing InfoProviders in the Join Control

Use

In order to define join conditions between two InfoProviders, you can bring up a new display screen separately and edit there.

To get an overview using the InfoProviders contained in the InfoSet, we recommend using the join control for the *Change InfoSet* screen. For this display a zoom factor of 50%, for example, would be suitable.

To edit individual InfoProviders within the InfoSet, we recommend using the separate display of two InfoProviders in the join control for the *Editing Selected Objects* screen. For this display a zoom factor of 120%, for example, would be suitable.

Prerequisites

You have transferred the InfoProviders you want from the *Change InfoSet* screen into the join control. For more information see Editing InfoSets [Page 179].

Procedure

1. You are in the *Change InfoSet* screen in the join control. Press the buttons *CTRL* + *Shift*, and select the two InfoProviders you want.
2. Choose *Selected Objects*. The *Editing Selected Objects* screen appears. The system displays both InfoProviders in full size.
3. Set or delete the join conditions you want.

   The following functions are available from the context menu (right mouse-click) of an entry in a table:

   - Hide Time-Dependent Fields
   - Left Outer Join or Inner Join
   - Select All Fields
   - Deselect All Fields

   The following editing functions are available by using buttons in the toolbar:

   Zoom in

   Zoom out

   Show/Hide Navigator

   Save as jpg

   Print

   For more information see Join-Control [Page 187].
4. Go back. You get to the *Change InfoSet* screen.

Result

You have edited two InfoProviders from your InfoSet. The system transfers the changes you made in the *Editing Selected Objects* screen into the display of the changed InfoProviders in the *Change InfoSet* screen.

If you choose *Back*, the system saves your personalized InfoProvider display settings in both join controls. Belonging to this would be, for example, the zoom size of both windows and the setting regarding whether the navigator is shown or hidden.

4.1.6.7.5.10 Temporal Join

Use

You use a temporal join to map a period of time.

During reporting, other InfoProviders handle time-dependent master data in such a way that the record that is valid for a predefined unique key date is used each time. InfoSets, however, are more flexible. They can be used to map periods of time, as in the following case:

A DataStore object contains a posting date and a time-dependent characteristic, as well as a key figure. You now want the record for the time-dependent characteristic to be determined according to the posting date, which is different in each record of the DataStore object. InfoSets enable this using temporal operands.

Features

A temporal join is a join that contains at least one time-dependent characteristic or a pseudo time-dependent InfoProvider.

In most cases, it makes sense to use one temporal operand for each InfoSet. This is because the key date check is carried out for each record of the results set, and for all temporal operands.

Temporal Operands

Temporal operands are time characteristics, or characteristics of type Date, for which an interval or a key date is defined. They influence the results set in the temporal join.

Key Date

In the *Key Date* column of the display in the join control [Page 187], you can set an indicator for these fields and attributes of an InfoProvider. If the indicator is set, the field or attribute is used as a temporal operand.

Depending on the type of characteristic, there are various ways to define a key date:

Characteristics of type Date and time characteristic 0CALDAY can be flagged as key dates.

You have multiple options for time characteristics that describe a period of time with a start and end date (0CALWEEK, 0CALMONTH, 0CALQUARTER, 0CALYEAR, 0FISCPER, 0FISCYEAR).

- *use first day as a key date*

- *use last day as a key date*
- *use a fixed day as a key date* (a particular day from the specified period of time)
- *key date derivation type:* You can specify a key date derivation type that you have defined using *Environment → Key Date Derivation Type.*

Time Interval

You can set time intervals for time characteristics that describe a period of time with a start and end date. Start and end dates are derived from the value of the time characteristic.

In the context menu of the table display of the InfoProvider, choose *Define Time-Dependency.* The system adds extra attributes (additional fields) to the relevant InfoProvider. These receive the start and end dates *from* (0DATEFROM) and *to* (0DATETO).

Pseudo Time Dependency of DataStore Objects and InfoCubes

In BI, only master data can be defined as a time-dependent data source. Two additional fields/attributes are added to the characteristic.

DataStore objects and InfoCubes themselves cannot be defined as time-dependent. However, they often contain time characteristics from which a time interval can be derived, or date entries with which you can define a time interval so that the corresponding InfoProvider in the InfoSet can be considered as time-dependent. The time characteristics 0CALWEEK, 0CALMONTH, 0CALQUARTER, 0CALYEAR, 0FISCPER, and 0FISCYEAR are considered in time derivation.

You can define pseudo time dependency in the following ways:

- Choose one of the previously mentioned time characteristics contained in the InfoProvider that is to be made time-dependent. Two date attributes are added to the InfoProvider in the InfoSet, and this indicates time-dependency.

  Example: If 0CALYEAR is derived with the value 2004, the start date has the value 01/01/2004 and the end date has the value 12/31/2004

- Flag a characteristic of type Date as the start date and another characteristic of type Date as the end date.

  You must make sure that the dataset is suitable for this. The value of the attribute that is interpreted as the start date must be smaller than, or equal to, the value of the attribute that is interpreted as the end date. If this is not the case, the data record is interpreted as invalid from the outset and is not taken into account in requests.

As soon as an InfoProvider contained in the InfoSet is made pseudo time-dependent, it is treated as a proper time-dependent data source.

An important difference between pseudo time-dependent InfoProviders and proper time-dependent InfoProviders is that the system cannot prevent gaps or overlaps from occurring in the time stream. This always depends on the dataset of the pseudo time-dependent InfoProvider.

Time Selection with Query Definition

A period of time is usually mapped for a temporal join. When defining queries, the question arises of how to restrict one or more key dates, or a combination of these, to a particular time interval. For technical reasons, it is not possible to define restrictions directly for fields *valid from* (0DATEFROM) and *valid to* (0DATETO) for the individual characteristics or the results set. For this reason, a dimension *valid time interval* (VALIDTIMEINTERVAL) exists for each InfoSet that represents a temporal join. This is only visible in the Query Designer and is used for the time selection.

Note the different ways in which the phrase *time interval* is used:

The **time interval** for a time-dependent InfoObject describes the period of time for which the respective record of the InfoObject is valid.

The InfoObjects for the time interval (*valid from* and *valid to*) of a time-dependent InfoObject are visible in the join control. If you set the indicator in the *Use Field* column, these fields are available in the BEx Query Designer to define a query, but cannot be restricted.

More information: Join Control [Page 187]

The **valid time interval** for a temporal join describes the period of time for which a record of the results set of the join is valid, and contains the following fields:

- *Valid from* and V*alid to:* These fields contain the beginning and the end of the valid time interval. They are **not** visible in the join control, but are available in the BEx Query Designer. These fields can only be used for the output of results in rows or columns. They must not be used with restrictions.
- *Time Interval:* This field is only used to select the time interval and can therefore only be used in the filter, but not to display results in rows or columns. The runtime system derives the correct selections for the database access from the *Time Interval* field. You can use multiple key dates and intervals as filters in the query definition. Temporal joins therefore enable you to display statuses for several times or time intervals next to each other in a query.

  Note: Restricting the time interval from 01.01.2001 to 31.12.2001 does not mean that the fields *valid from* and *valid to* take these values. Instead, this restriction results in every record of the results set having a validity area that lies either entirely or partially within this time interval.

Time Dependency in the Results Set

Time-dependency is assessed when the results set is determined. A record is only included in the results set if the key date or time interval lies within the valid time interval. A time interval is assigned to each record in the results set. The records are valid for the duration of the interval to which they are assigned (valid time interval).

Since a key date or a time interval can only be derived from a time characteristic once the results set has been read, the system checks the validity of the records again after the data has been read from the database. As a result, more data is read than ultimately appears as the query result. You must therefore think about the effect on the system performance before you use time characteristics as temporal operands with derivations.

It is much better for performance to calculate and fill two date fields (start and end date) from the derived time characteristic during data loading. You can then define these fields in the InfoSet as start and end date.

Example: A DataStore object or an InfoCube has the time characteristic 0CALMONTH. This is to be used later in the InfoSet as a time interval, and therefore the InfoCube or the DataStore object should be considered as pseudo time-dependent. You insert two fields of type Date (Date_01, Date_02) into the DataStore object or InfoCube and fill them when loading.

If 0CALMONTH has the value 092004, the fields are filled as follows:

Date_01 → 09/01/2004, Date_02 → 09/30/2004

If you use Date_01 and Date_02 as interval limits, the SQL statement takes them into account. The results set is therefore much more likely to be smaller than if you were to execute the derivation using 0CALMONTH.

You are however able to use an InfoObject of data type D and InfoObject 0CALDAY as temporal operands without restriction. This is because the corresponding selection conditions are directly relayed to the database.

If only **one time-dependent characteristic** is contained in the join, note that there are multiple records in the database for a value of this characteristic. For this reason, multiple records can appear in the results set for the join; they can be only be distinguished from one another by time-dependent attributes and the valid time interval of the characteristic. You can filter such records using a time selection. For more information, see the third example in Interpreting Queries Using InfoSets [Page 208].

If **two time-dependent characteristics** are contained in the join, only those combinations of InfoObject records that have a common validity area regarding the time period are included in the results set. This also applies if there are more than two time-dependent InfoObjects in a join.

For example, a join contains the following time-dependent InfoObjects (in addition to other objects that are not time-dependent):

InfoObjects in the Join	Valid From	Valid To
Cost center (0COSTCENTER)	01/01/2001	05/31/2001
Profit center (0PROFIT_CTR)	03/01/2001	07/31/2001

Where the two time intervals overlap, that is, the validity area that the InfoObjects have in common, is the valid time interval of the temporal join:

Temporal Join	Valid From	Valid To
Valid time interval	03/01/2001	05/31/2001

You define an InfoSet using the PROFITC (profit center) characteristic. This contains the responsible person (RESP) as the time-dependent attribute and the CSTSNTR (cost center) characteristic, which also contains the person responsible as a time-dependent attribute. These characteristics contain the following records:

PROFITC	RESP	DATEFROM	DATETO
BI	John Smith	01/01/2000	06/30/2001
BI	Jane Winter	07/01/2001	12/31/9999

CSTCNTR	PROFITC	RESP	DATEFROM	DATETO
4711	BI	Sue Montana	01/01/2001	05/31/2001
4711	BI	Peter Street	06/01/2001	12/31/9999
4711	BI	Dan Barton	01/01/2002	12/31/9999

If both characteristics are used in a join and are connected using PROFITC, not all six possible combinations are valid for the above records, but only the following four:

PROFITC	RESP	CSTCNTR	PROFITC	RESP	
BI	John Smith	4711	BI	Sue Montana	(01/01/2001-

					05/31/2001)
BI	John Smith	4711	BI	Peter Street	(06/01/2001-06/30/2001)
BI	Jane Winter	4711	BI	Peter Street	(07/01/2001-12/31/2001)
BI	Jane Winter	4711	BI	Dan Barton	(01/01/2002-12/31/9999)

The valid time interval for the combinations, that is, the time period in which the records of both characteristics are valid, is displayed in parentheses. The combinations of responsible persons John Smith and Dan Barton, or Jane Winter and Sue Montana, are not allowed, as their validity areas do not overlap.

More information:

Processing the Time Dependency [Page 203]

4.1.6.7.5.10.1 Processing the Time Dependency

In the case of a time-dependent InfoSet, the system processes the conditions as follows:

Exclude conditions on the valid time characteristic are always converted into include conditions. This type of processing guarantees the display of correct results.

See SAP Note 1043011, which describes the behavior in more detail. Before SAP NetWeaver 7.0, SPS 12, the conditions were processed differently. The SAP Note also describes how to reset the behavior.

The following examples clarify how the conditions are converted:

Examples for Single Values:

The condition

I	EQ	20000228

is converted to

I	LE	20000227
I	GE	20000229

The condition

I	NE	20000228

is converted to

I	LE	20000227
I	GE	20000229

The condition

E	LT	20000228

is converted to

I	GE	20000228

The condition

E	LE	20000228

is converted to

I	GE	20000229

The condition

E	GT	20000228

is converted to

I	LE	20000228

The condition

E	GE	20000228

is converted to

I	LE	20000227

The condition

E	NE	20000228

is converted to

I	EQ	20000228

Examples for Intervals:

The condition

E	BT	20000228	20000331

is converted to

I	LE	20000227
I	GE	20000401

The condition

E	NB	20000228	20000331

is converted to

I	BT	20000228	20000331

The condition

I	NB	20000228	20000331

is converted to

I	LE	20000227
I	GE	20000401

4.1.6.7.5.11 Most Recent Reporting for InfoObjects

Use

This function allows you to report on the master data last loaded into the system, even when they haven't been activated yet.

Functions

You can find this function in the InfoSet Builder via the main menu under *Goto → Global Properties → Most recent Reporting for InfoObjects.* If you set this indicator, a most recent reporting is carried out for all master-data bearing characteristics. As a result, the newest records are displayed in the query, even when these have yet to be activated, that is, are still in the M version. See Versioning Master Data [Page 120].

Example

You have defined an InfoSet via the characteristic 0COSTCENTER. You are loading master data for the 0COSTCENTER characteristic. After activation, the P table looks like this:

CO_AREA	COSTCENTER	OBJVERS	CHANGED	RESP_PERS
1000	0000001000	A		Patel
1000	0000001020	A		Jones

Later, you load new records. These are firstly in the M version:

CO_AREA	COSTCENTER	OBJVERS	CHANGED	RESP_PERS
1000	0000001000	A		Patel
1000	0000001020	A	D	Jones
1000	0000001020	M	I	Smith

In a query based on this InfoSet without the function *Most recent Reporting*, all active data records are considered:

CO_AREA	COSTCENTER	OBJVERS	CHANGED	RESP_PERS
1000	0000001000	A		Patel
1000	0000001020	A		Jones

The newest data records are considered in a query on this InfoSet **with** the function *Most recent Reporting*:

CO_AREA	COSTCENTER	OBJVERS	CHANGED	RESP_PERS
1000	0000001000	A		Patel
1000	0000001020	M	I	Smith

4.1.6.7.5.12 Local Aggregation

Use

In the global properties for an InfoSet, you can specify whether a local aggregation it to be used.

This setting is used to control how the key figures are grouped.

- If the indicator is set, the key figures are grouped before the join, for all tables involved.
- If the indicator is not set, grouping does not take place until after the table contents are joined. This is the default setting.

This setting affects the query results.

Example

A DataStore object, 'ZZSALES', contains the following data records:

#CUSTOMER	#MATERIAL	#COUNTRY	QUANTITY	PRICE
MILLER	10001	US	5	500 USD
MILLER	10001	DE	7	700 USD
MILLER	10002	US	3	150 USD
MILLER	10002	DE	2	100 USD

Another DataStore object, 'ZZMATERIAL', contains the following data records and provides an overview of the materials and quantities available in the different storehouses:

#MATERIAL	#STOREHOUSE	QUANTITY
10001	11	5
10001	12	4
10001	13	2
10002	11	20
10002	12	10
10002	13	0

In the InfoSet, the two InfoProviders are joined:

ZZSALES-MATERIAL with ZZMATERIAL-MATERIAL

In the query, the fields ZZSALES-CUSTOMER, ZZSALES-MATERIAL, ZZSALES-QUANTITY, ZZSALES-PRICE and ZZMATERIAL-QUANTITY, for example, are displayed. The result is as follows:

1. without grouping/aggregation

#CUSTOMER (ZZSALES)	#MATERIAL (ZZSALES)	QUANTITY (ZZSALES)	PRICE (ZZSALES)	QUANTITY (ZZMATERIAL)
MILLER	1001	5	500 USD	5
MILLER	1001	7	700 USD	5
MILLER	1001	5	500 USD	4
MILLER	1001	7	700 USD	4
MILLER	1001	5	500 USD	2
MILLER	1001	7	700 USD	2
MILLER	1002	3	150 USD	20
MILLER	1002	2	100 USD	20
MILLER	1002	3	150 USD	10
MILLER	1002	2	100 USD	10
MILLER	1002	3	150 USD	0
MILLER	1002	2	100 USD	0

Each row from the ZZSALES table appears three times in the result, because it has three rows as join partners in the ZZMATERIAL table. Similarly, each row from the ZZMATERIAL table appears twice in the result, because it has two rows as join partners in the ZZSALES table.

2. without local grouping/aggregation before the join (indicator is not set)

The result is grouped/aggregated for all characteristics:

#CUSTOMER (ZZSALES)	#MATERIAL (ZZSALES)	QUANTITY (ZZSALES)	PRICE (ZZSALES)	QUANTITY (ZZMATERIAL)
MILLER	1001	36	3600 USD	22
MILLER	1002	15	750 USD	60

This result does not necessarily match the expected result from a business perspective. If the join condition of the tables involved does not fully cover the entire key of both tables, it is possible that multiple data records from a table are found that satisfy the join condition. In this case, the join is not unique and the key figure values are duplicated.

3. with local grouping/aggregation before the join (indicator is set)

To avoid duplication of key figure values, data from each table involved can be grouped/aggregated before the join is performed. This ensures that the results are as expected and do not include any duplicate records.

The tables involved are first aggregated. The result for ZZSALES is:

#CUSTOMER (ZZSALES)	#MATERIAL (ZZSALES)	QUANTITY (ZZSALES)	PRICE (ZZSALES)
MILLER	1001	12	1200 USD
MILLER	1002	5	250 USD

The result for ZZMATERIAL is:

#MATERIAL (ZZMATERIAL)	QUANTITY (ZZMATERIAL)
1001	11
1002	30

The join is then made and the end result is:

#CUSTOMER (ZZSALES)	#MATERIAL (ZZSALES)	QUANTITY (ZZSALES)	PRICE (ZZSALES)	QUANTITY (ZZMATERIAL)
MILLER	1001	12	1200 USD	11
MILLER	1002	5	250 USD	30

4.1.6.7.5.13 Interpreting Queries Using InfoSets

The following explanations and examples are intended to help you understand how queries that use InfoSets work, and to be able to interpret the results correctly.

Technical Issues That Affect the Result of the Query

The results set of a join is made up of fields from all of the tables involved. One row of this result set contains a valid combination of rows from each of the tables involved. The join condition and the filter for the query that you specify determine which combinations are valid.

You can set join conditions between fields from the key part of the tables and between fields from the data part of the tables. For two InfoObjects, for example, you can define an equal join condition between two attributes.

The filter for the query determines which values are allowed for individual columns of the results set, or the combinations of values that are allowed for various different columns. This further restricts the results set that is produced by the join condition.

Depending on how join conditions have been designed, every record from table1 and table2 can be included several times in a combination for a record in the results set.

For example, if for a single record in table1 there are a total of three records in table2 for which the conditions F1(T1) = F2(T2) apply, there are potentially three records in the results set in which the record from table1 is included. If table1 contains a key figure, depending on the filter condition in place, this key figure can appear one to three times or not at all in the results set. The data for the query is determined from the results set.

First of all, the data is compressed using the characteristics that you do not want to be displayed in the query. Different values for the same key figure can be output for the same combinations of characteristics in various queries, which can result in different totals.

Therefore, you should note the *Number of Records* key figure. This is included in every InfoSet. This key figure tells you how many records in the results set for the join feed into a record in the query.

Example

You are using the following objects in a scenario:

- DataStore object DS_ORDER

Key: ORDER_NO

Data part: PERSON, PLANT, AMOUNT, ...

- Characteristic PLANT (time independent)

  Key: PLANT

  Data part (attribute): ...

- Characteristic PERSON (time dependent)

  Data part (attribute): ...

- Characteristic BPARTNER (time independent)

  Key: BPARTNER

  Data part (attribute): PLANT: ...

In the following examples it is assumed that master data exists for all of the characteristics in the data part of DS_ORDER. (Otherwise you would have to work with a left outer-join.)

1. An InfoSet contains a join from DataStore object DS_ORDER and characteristic PLANT. You have defined the join condition PLANT(DS_ORDER) = PLANT(PLANT). In this example, for each record in DS_ORDER, there is exactly one record in PLANT. The AMOUNT key figure cannot be included more than once in the results set.
2. An InfoSet contains a join from DataStore object DS_ORDER and characteristic BPARTNER. You have defined the join condition PLANT(DS_ORDER) = PLANT(BPARTNER). A number of records from BPARTNER may have the same value for PLANT. This means that more than one record from BPARTNER may be determined for a single record in DS_ORDER. As a result, there is more than one record in the result set of the join and the AMOUNT key figure appears several times. You can avoid this by activating local aggregation. For more information, see Local Aggregation [Page 206].
3. An InfoSet contains a join from DataStore object DS_ORDER and time-dependent characteristic PERSON. You have defined the join condition PERSON(DS_ORDER) = PERSON(PERSON). Although physically a person is unique and can exist only once, the fact that the PERSON characteristic is time-dependent means that several records can exist for a single person. Using time-dependent characteristics results in a situation like that described in the second example. Note the time selection options [Page 199] that are available for time-dependent characteristics in temporal joins which allow you to avoid this type of situation.

4.1.6.7.5.14 Classic InfoSet

Definition

A classic InfoSet provides a view of a dataset that you report on. You use the InfoSet query for this purpose. The classic InfoSet determines which tables, or fields within a table, an InfoSet query references.

Use

As of Release BW 2.0B, InfoSets are used in the Business Information Warehouse for InfoObjects (master data), DataStore objects, and joins for these objects. These InfoSets are not BI Repository objects but SAP Web Application Server objects. The InfoSet query can be used to carry out tabular (flat) reporting on these InfoSets.

As of Release BW 3.0, these InfoSets are called **classic InfoSets.**

Integration

As of Release BW 3.0A, you can use **transformation program RSQ_TRANSFORM_CLASSIC_INFOSETS** to avoid having to fully implement classic InfoSets that were constructed and used in Release BW 2.0B. This simplifies the procedure. All DataStore objects and all InfoObjects, as well as their join conditions, are transferred into the new object.

The following restrictions apply:

For technical reasons, all additional definitions of the classic InfoSet (additional tables, additional fields, text fields, limits, coding for the various points in time) are **not** transferred into the new InfoSet. Comparable definition options are not available in the new InfoSets. Alternatively, use the option available when defining BEx queries (calculated key figures, for example). For more information, see Query Design: BEx Query Designer [External]. If this method is not a sufficient replacement for the definitions stored in a classic InfoSet, continue to use the classic InfoSet.

You **cannot** transform InfoSet queries.

1. In the ABAP Editor, start program RSQ_TRANSFORM_CLASSIC_INFOSETS (transaction SE38). The *Classic InfoSet → InfoSet Conversion* screen appears.
2. On the selection screen, enter the name of a classic InfoSet in the system, as well as the name of an InfoSet that is not already in the system.
3. Choose *Execute*. The program checks whether transformation is possible, carries it out if necessary, and then activates the newly created InfoSet.

4.1.6.7.5.14.1 Setting-up a Role for the InfoSet Query

Use

To be able to create an InfoSet Query, the system administrator must set up a role for the work with the InfoSet Query.

Procedure

1. A role has to be assigned specifically to a single SAP Query user group. This is because the InfoSet Query is derived from the SAP Query.
   a. Call up the *Role Maintenance* (RSQ10). You get to a table containing the roles that are relevant for working with the InfoSet Query.
   b. Choose the role you want, and use the *Assign User Group* function (second column in the table) to assign a user group to the role, or to remove an existing assignment.

      When you are assigning user groups, a dialog box appears asking you if you want to create a new user group, or use an existing one. Use the input help to choose from the available user groups. It is not possible to assign a user group to more than one role.

      When you have assigned a user group successfully to a role, the name of this user group appears in the third column of the table.

It is also possible to jump to the Query Builder from the SAP Easy Access SAP Business Information Warehouse Menu by selecting the *InfoSet Query* entry from the roles in the user menu.

2. Assign Classic InfoSets to the role. Use the *Assign Classic InfoSets* function to do this (fifth column in the table).

   A screen containing all the available Classic InfoSets appears. Select the InfoSets that you want to be able to use for defining queries within the role.

   You are able to choose one of the selected Classic InfoSets as a standard Classic InfoSet (entry in the fourth column of the table). The standard Classic InfoSet is subsequently used as a template, if the components for maintaining InfoSet Queries are called using the menu entry mentioned above.

3. In the *Maintain Role* transaction (PFCG) you assign the role you have set up to those users, who are going to work with the InfoSet Query.

See also:

Process Classic InfoSets and Assign Roles [Page 211]

Creating InfoSet Queries [Page 213]

4.1.6.7.5.14.2 Processing Classic InfoSets and Assigning Roles

Use

Before you are able to work with the classic InfoSet query, classic InfoSets must already be available that are assigned to particular roles.

Procedure

There are various ways of getting to the classic InfoSet maintenance screen:

- Call transaction RSQ02 *InfoSet: Initial Screen*.
- Call from the context menu of DataStore objects or InfoObjects in the *Modeling* view of the *Data Warehousing Workbench*

  The following deals primarily with BI-specific enhancements. You can find extensive information about the available functions of SAP Application Server InfoSets in the SAP documentation on SAP Query. This information also covers BI classic InfoSets.

Defining Classic InfoSets

1. In the *Data Warehousing Workbench – Modeling*, choose the *Classic InfoSets* function from the context menu of the object that requires a classic InfoSet. In the right-hand side of the screen, the classic InfoSets are displayed that use this particular object. In the classic InfoSet overview, you can access the most important functions (*Change*, *Display*, *Delete*, and so on) from the transaction (RSQ02) in the *Classic InfoSet* menu. To see if any queries already exist for a classic InfoSet, choose the *Query Directory* function. This lists all the queries that have already been created for a classic InfoSet. The *Classic InfoSet Maintenance* option takes you to the initial screen of *Classic InfoSet Maintenance*. All the existing classic InfoSets are listed here.
2. Choose one of the following functions to create a new classic InfoSet for a particular object:

- *Recreate Standard Classic InfoSets*.

  A classic InfoSet that contains all attributes is created for InfoObjects.

  For DataStore objects, the system generates a classic InfoSet from the table of active data and a “new and active data combined” classic InfoSet. If there is a more up-to-date record in the table of new data, this record is used in reporting instead of the active record. You can modify the standard Classic InfoSet. Note that only the generated version can be used for the InfoSet query.

- If you want to use joins, you have to define the classic InfoSet manually. Specify a name in the *Technical Name* field and choose *Create New Classic InfoSet*.

  The *Classic InfoSet Maintenance* screen appears The *Classic InfoSet: Title and Data Source* screen appears. The system has already identified the appropriate basic master data table or DataStore object table as the data source.

  If you want to use a classic InfoSet for queries in a Web environment, you have to assign the InfoSet to a user group. Do this in the *Classic InfoSet: Title and Data Source* screen. To call this screen in the future, choose the *Global Properties* function from the *Goto* menu in the *Classic InfoSet Maintenance* screen.

  After you confirm you entries, *Join Definition* is displayed. Determine the join conditions.

  In the main screen of the *Classic InfoSet Maintenance*, you can:

  - Choose all the attributes you require
  - Arrange these attributes into field groups
  - Determine the fields that are going to contain extra information (characteristics and key figures, for example) that was not contained originally in the InfoObjects or DataStore objects.

3. Save your entries.

   The *Settings* function in the *Classic InfoSet Maintenance* screen allows you to switch to using DDIC names. You use this option, for example, when you are writing coding, defining upper and lower limits for a classic InfoSet, or connecting additional tables, and you have to give the DDIC names rather than the technical names used in the BI system.

Assigning Classic InfoSet to a Role

Once you have saved and generated the classic InfoSet, you assign it to one or more roles. Choose the *Role Assignment* function in the classic InfoSet overview. All the roles that have been set up for working with the InfoSet query are displayed in a dialog box.

The roles that the classic InfoSet is already assigned to are highlighted in the first column. Use the corresponding field to:

- Assign the classic InfoSet to another one of the roles
- Delete an existing assignment. Before you do this, make sure that there are no queries left for the classic InfoSet within this particular role.

Result

You can now use the classic InfoSet in the Query Builder, within the context of the role assigned to it.

See also:

Creating InfoSet Queries [Page 213]

4.1.6.7.5.14.3 InfoSet Query

The InfoSet Query is a tool of the SAP Application Server that can be used in the BI System for tabular (flat) reporting. Classic InfoSets provide the view for this data.

Use

The InfoSet Query versus the BEx Query

The BEx Query is intended for reporting with InfoCubes. In this process, functions such as the following are supported: Reporting with all InfoProviders, using variables, navigation, displaying attributes and hierarchies, characteristic drilldown, currency translations, report-report interface (R-RI), and authorization checks. These functions are also available for reporting using DataStore objects. However, there is a restriction in that you cannot report on more than one DataStore object at a time. The tool designed for this is the InfoSet Query.

The InfoSet Query is designed for reporting using flat data structures, that is InfoObjects, DataStore objects, and DataStore object joins. The following functions are supported for the InfoSet Query: Joins from several master data tables and DataStore objects, report-report interface (R-RI), and authorization checks. The authorization check in the InfoSet Query is simpler than the authorization check in the BEx query. The report is displayed either in the SAP List Viewer, or on the Web.

Constraints

We recommend you do **not** use the InfoSet Query for reporting using InfoCubes. The InfoSet Query does **not** support the following functions: Navigation, hierarchies, delivery of BI Content, currency translation, variables, exception reporting, and interactive graphics on the Web.

4.1.6.7.5.14.4 Creating InfoSet Queries

Use

The InfoSet Query is designed for reporting on data stored in flat tables. It is particularly useful for reporting on joins for master data and joins for DataStore objects.

Prerequisites

You must take the following steps before you can create Infoset queries:

- Set up Roles for the InfoSet Query [Page 210]

- Process Classic InfoSets and Assign Roles [Page 211]

Procedure

Define the InfoSet Query

1. Call the Query Builder. There are various ways of doing this:
   - To call the Query Builder from the corresponding role menu or from the BEx Browser, double-click on *InfoSet Query* in the menu that is created when you set up a role.
   - Developers and testers of Classic InfoSets are able to call up the Query Builder directly from the Classic InfoSet overview in the Data Warehousing Workbench.

   If several Classic InfoSets are assigned to a role, and one of them has been identified as a standard Classic InfoSet, this Classic InfoSet is used as a template when the query is called up. To change the template, choose *Create New Query – Classic InfoSet Selection*. Any of the Classic InfoSets that are assigned to the role can be the new template.
2. Define your query. The procedure is similar to the procedure for defining queries in the BEx Analyzer.

   Transfer individual fields from the field groups you have selected in the Classic InfoSet into the preview. To do this, use the drag and drop function, or highlight the relevant fields in the field list.

   Use either of these two methods to select any fields you want to use as filters. These fields are displayed in the *Selections* area of the screen (top right).

   When you are preparing the query, only example data is displayed in the *Preview*. When you choose the *Output* or *Refresh* function, the actual results are displayed on the same screen.
3. Execute the query.
4. Choose from the following options:

   Ad hoc reporting

   You do not want to save the query for later. Save the Query Builder without saving.

   Reusable queries

   You want to save the query, because you want to work on it later, or use it as a template. Use either the *Save* or the *Save as* function to save the query.

   In addition to the Classic InfoSets that you assigned to the role, you are also able to use the query as a template. It is not possible, however, to access the query from other roles.

   After you save the query, a second dialog box appears, asking you if you want to save the query as a separate menu entry within the role. If you choose this option, you are able to start the query directly from the user menu or from the BEx Browser. It is also possible to use the *Role Maintenance* transaction (PFCG) to save this kind of role entry.

   Choose *Menu* → *Refresh* to display the query.

   If you want to change or delete the saved query, use the *Edit* function from the context menu in the BEx Browser to call the maintenance tool for InfoSet Queries with this query as a template.

InfoSet Query on the Web

It is possible to publish each InfoSet Query on the Web. There are the following display options:

- MiniALV for creating MiniApps in the SAP Workplace
- MidiALV without selection options
- MidiALV with selection options

Both the MiniALV and the MidiALV allow you to switch between various selection/layout variants. The publishing screen for the data is adjusted individually using URL parameters.

The following prerequisites are necessary for security reasons:

- Releasing the query for the Web
- Specifying an authorization group for the corresponding Classic InfoSet

  Call up transaction RSQ02 *InfoSet: Entry*, and choose *Go to* → *Additional Functions* → *Web Administration of Queries*. Make the corresponding entries.

4.1.6.7.6 MultiProviders

Definition

A MultiProvider is a type of InfoProvider that combines data from a number of InfoProviders and makes it available for analysis purposes. The MultiProvider itself does not contain any data. Its data comes entirely from the InfoProviders on which it is based. These InfoProviders are connected to one another by a union operation.

Use

A MultiProvider allows you to analyze data based on several InfoProviders.

See the following examples:

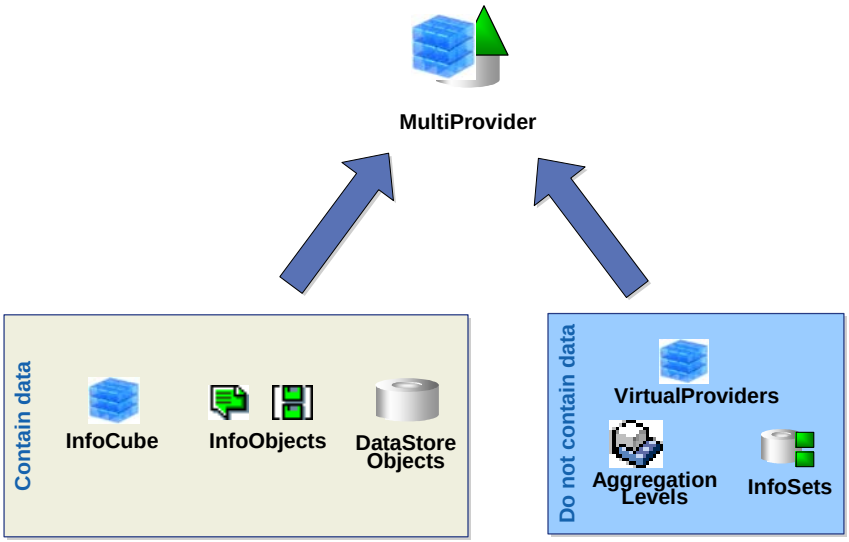
Example: List of Slow-Moving Items [External]

Example: Plan-Actual Data [Page 220]

Example: Sales Scenario [Page 221]

Structure

A MultiProvider can consist of different combinations of the following InfoProviders: InfoCube, DataStore object, InfoObject, InfoSet, VirtualProvider, and aggregation level.



A union operation is used to combine the data from these objects in a MultiProvider. Here, the system constructs the union set of the data sets involved; all the values of these data sets are combined. As a comparison: InfoSets are created using joins. These joins only combine values that appear in both tables. In contrast to a union, joins form the intersection of the tables

As a comparison, see InfoSet [Page 176].

In a MultiProvider, each characteristic in each of the InfoProviders involved must correspond to exactly one characteristic or navigation attribute (where these are available). If this is not clear, you have to specify the InfoObject to which you want to assign the characteristic in the MultiProvider. You do this when you define the MultiProvider.

The MultiProvider contains the characteristic 0COUNTRY and an InfoProvider contains the characteristic 0COUNTRY as well as the navigation attribute 0CUSTOMER__0COUNTRY. In this case, select just one of these InfoObjects in the assignment table.

If a key figure is contained in a MultiProvider, you have to select it from (at least) one of the InfoProviders contained in the MultiProvider. In general, one InfoProvider provides the key figure. However, there are cases in which it is better to select the key figure from more than one InfoProvider:

If the 0SALES key figure is stored redundantly in more than one InfoProvider (meaning that it is contained fully in all the value combinations for the characteristics), we recommend that you select the key figure from just one of the InfoProviders involved. Otherwise the value is totaled incorrectly in the MultiProvider because it occurs several times.

However, if 0SALES is stored as an actual value in one InfoProvider and as a planned value in another InfoProvider and there is no overlap between the data records (in other words, sales are divided separately between several InfoProviders), it is useful to select the key figure from more than one InfoProvider.

Integration

MultiProviders only exist as a logical definition. The data continues to be stored in the InfoProviders on which the MultiProvider is based.

A query based on a MultiProvider is divided internally into subqueries. There is a subquery for each InfoProvider included in the MultiProvider. These subqueries are usually processed in parallel.

The following sections contain more detailed information:

Dividing a MultiProvider Query into Subqueries [Page 219]

Processing Queries [External]

Technically there are no restrictions with regard to the number of InfoProviders that can be included in a MultiProvider. However, we recommend that you include no more than 10 InfoProviders in a single MultiProvider, otherwise splitting the MultiProvider queries and reconstructing the results for the individual InfoProviders takes a substantial amount of time and is generally counterproductive. Modeling MultiProviders with more than 10 InfoProviders is also highly complex.

See also:

Recommendations for Modeling MultiProviders [Page 16]

4.1.6.7.6.1 Creating MultiProviders

Prerequisites

There is an active version of each InfoObject that you want to transfer to the MultiProvider. Create any InfoObjects that you require that do not already exist and activate them.

Instead of creating a new MultiProvider, you can install a MultiProvider from SAP Business Content [Page 435].

Procedure

5. Create an InfoArea to which you want to assign the new MultiProvider.

   Choose *Modeling* → *InfoProvider*.

6. In the context menu of the InfoArea, choose *Create MultiProvider*.
7. Enter a technical name and a description.
8. Choose *Create.*
9. Select the InfoProvider that you want to form the MultiProvider. Choose *Continue*. The *MultiProvider* screen appears.
10. Use drag and drop to transfer the required InfoObjects into your MultiProvider. You can also transfer entire dimensions.
11. Use *Identify Characteristics* and *Select Key Figures* to make InfoObject assignments between MultiProviders and InfoProviders.

In a MultiProvider, each InfoObject in the MultiProvider must correspond to exactly one InfoObject in each of the InfoProviders involved (as long as it is available in the MultiProvider). If this mapping is not clear, you have to specify the InfoObject to which you want to assign the InfoObject in the MultiProvider.

See also, Consistency Check for Compounding [Page 218].

12. *Save* or *Activate* the MultiProvider. Only active MultiProviders are available for analysis and reporting.

See also:

The additional functions in DataStore object maintenance [Page 158] are also available as additional functions in MultiProvider maintenance. The only exception is the last function listed for performance settings.

4.1.6.7.6.1.1 Consistency Check for Compounding

With regard to compounding, characteristics and navigation attributes have to be identified consistently within a MultiProvider. Otherwise the query results may be inconsistent. Data records may appear in the MultiProvider that do not physically exist in any of the InfoProviders in the MultiProvider.

The system checks for consistency. If a MultiProvider is not consistently modeled, it cannot be activated. The system produces an error message. However, you can change this to a warning. This allows you to activate the MultiProvider anyway. Only do this if you are certain that it will not result in inconsistent values.

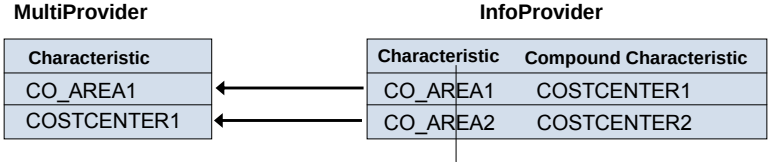
If you have upgraded from SAP BW 3.x to SAP NetWeaver 2004s, MultiProviders defined in SAP BW 3.x may be seen as being incorrect and can no longer be activated. In this case, check the definition and modify it as required. For more information about how to execute a check using a report, see MultiProvider [External].

Example: Inconsistent Compounding

Characteristic cost center 1 (COSTCENTER1) is compounded to characteristic controlling area (CO_AREA1). Characteristic cost center 2 (COSTCENTER2) references characteristic COSTCENTER1. Characteristic controlling area 2 (CO_AREA2) references characteristic controlling area 1 (CO_AREA1). As a result, COSTCENTER2 is compounded to CO_AREA2.

These four characteristics are contained in an InfoProvider and the MultiProvider.

The following graphic shows how the characteristics are identified:



COSTCENTER2 is mapped to COSTCENTER1. This means that the higher-level characteristic CO_AREA2 in the InfoProvider also has to be mapped to CO_AREA1 (because this is the higher-level characteristic for COSTCENTER1 in the MultiProvider). This is not the case, which means that the compounding is not consistent.

Correct the assignment:

CO_AREA1 ← CO_AREA2

The following example illustrates the problem with inconsistent assignments. You have the following master data:

CO_AREA	COSTCENTER
1000	A
2000	B
2000	C

Because of the incorrect assignment shown above, the following data record could be created in the MultiProvider.

CO_AREA	COSTCENTER
1000	C

This master data does not exist.

4.1.6.7.6.2 Dividing a MultiProvider Query into Sub-Queries

Use

A query based on a MultiProvider is divided internally into sub-queries. A sub-query is generated for each InfoProvider belonging to the MultiProvider.

Features

The division of a MultiProvider query into sub-queries can be very complex. If you have defined a query for a MultiProvider and want to see how the query has been sub-divided, call transaction RSRT. This can be a useful step if your query does not behave as expected.

To see how the query is divided, proceed as follows:

Use RSRT to execute the query with the *Execute + Debug* option. Choose the *Explain MultiProvider* option. The upper area of the screen, in which the query result is displayed, contains messages with information about how the query has been divided. You may see the following messages:

- DBMAN 133: There is a mapping rule that maps a characteristic (or navigation attribute) in the MultiProvider to a characteristic (or navigation attribute) of the same type (but not the same name) in the specified InfoProvider.
- DBMAN 134: The query contains a general restriction for the specified characteristic (or navigation attribute). This is not available in the specified InfoProvider. This is probably the reason why the sub-query is omitted from this InfoProvider.
- DBMAN 135: The specified key figure is either not available in the specified InfoProvider or it has not been selected for the MultiProvider. As a result, the sub-query does not read any values for this key figure.
- DBMAN 136: The sub-query for the selected InfoProvider has been excluded. The reasons for this are found in the preceding messages.
- DBMAN 137: A characteristic (or navigation attribute) is not available in the specified InfoProvider. For this reason, all the conditions in the same query column are irrelevant, and are not considered in the sub-query.

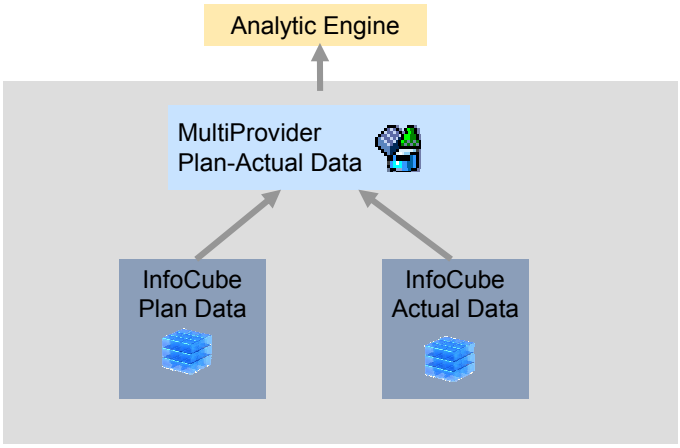
- DBMAN 138: All the conditions have been deleted for all the query columns (see DBMAN 137). This is because they could not be filled from the specified InfoProvider. Therefore the system does not access these key figures.
- DBMAN 139: The query only contains key figures that do not appear in the specified InfoProvider. Therefore the system does not access these key figures.
- DBMAN 140: A characteristic is set to a particular constant value for an InfoProvider. This condition is not consistent with a condition contained in the MultiProvider query. As a result, the system does not access the specified InfoProvider.
- DBMAN 141: This message describes a query restriction that was referred to in a previous message. It contains information about
  - the InfoCube or InfoProvider in question
  - the query column (FEMS)
  - whether the condition is inclusive (I) or exclusive (E)
  - the characteristic (or navigation attribute) involved
  - the relational operator
  - the operands of the condition (possibly)
- DBMAN 144: This message describes a situation in which a restriction for characteristic A in the MultiProvider can apply to characteristic B in the specified InfoProvider since a restriction (of the same level) already exists for characteristic B. The specified InfoProvider reads the data without this restriction. This restriction is processed subsequently by the OLAP processor.
- DBMAN 145: The specified InfoObject is interpreted as a real key figure for the specified InfoProvider. This can be relevant for a MultiProvider query when all other key figures in the query are not available in this InfoProvider and the sub-query would need to be excluded (see DBMAN 139). In this case, this option is not available.

See also:

Processing Queries [External]

4.1.6.7.6.3 Example: Plan-Actual Data

You have one InfoProvider with the actual data for a logically related business area and one equivalent InfoProvider with the plan data. To compare the actual data with the planned data in one query, you combine the two InfoProviders into one MultiProvider.



This is a homogeneous data model. Homogeneous MultiProviders consist of InfoProviders that are technically the same, for example, InfoCubes with exactly the same characteristics and similar key figures. In this case, the InfoCube with the plan data contains key figure *Planned Costs* and the InfoCube with the actual data contains key figure *Actual Costs*.

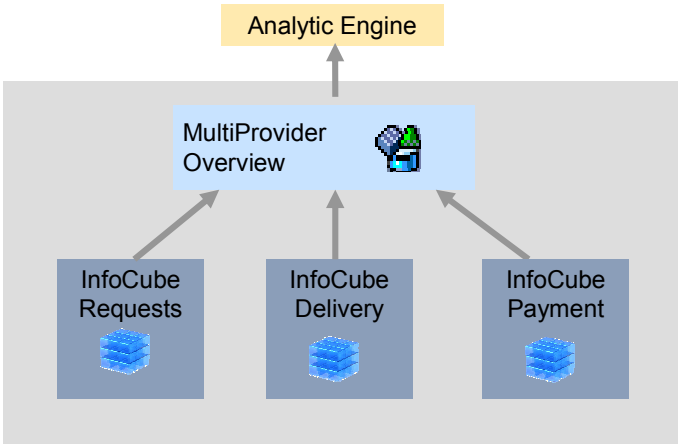
Homogeneous MultiProviders represent one way in which you can achieve partitioning within modeling.

4.1.6.7.6.4 Example: Sales Scenario

You want to model a sales scenario that is made up of the sub-processes *order*, *delivery* and *payment*. Each of these sub-processes has its own (private) InfoObjects (*delivery location* and *invoice number,* for example) as well as a number of cross-process objects (such as *customer* or *order number*). It makes sense here to model each sub-process in its own InfoProvider and then combine these InfoProviders into a MultiProvider. It is possible to:

- Model all sub-scenarios in one InfoProvider, or
- Create an InfoProvider for each sub-scenario, and then combine these InfoProviders into a single MultiProvider.

The second option usually simplifies the modeling process and can improve system performance when loading and reading data. There is one InfoCube for order, delivery and payment respectively. You can execute individual queries for the individual InfoCubes or obtain an overview of the entire process by creating a query based on the MultiProvider.



This is a heterogeneous data model. Heterogeneous MultiProviders are made up of InfoProviders that only have a certain number of characteristics and key figures in common. Heterogeneous MultiProviders can be used to simplify the modeling of scenarios by dividing them into sub-scenarios. Each sub-scenario is represented by its own InfoProvider.

4.1.6.8 Open Hub Destination

Definition

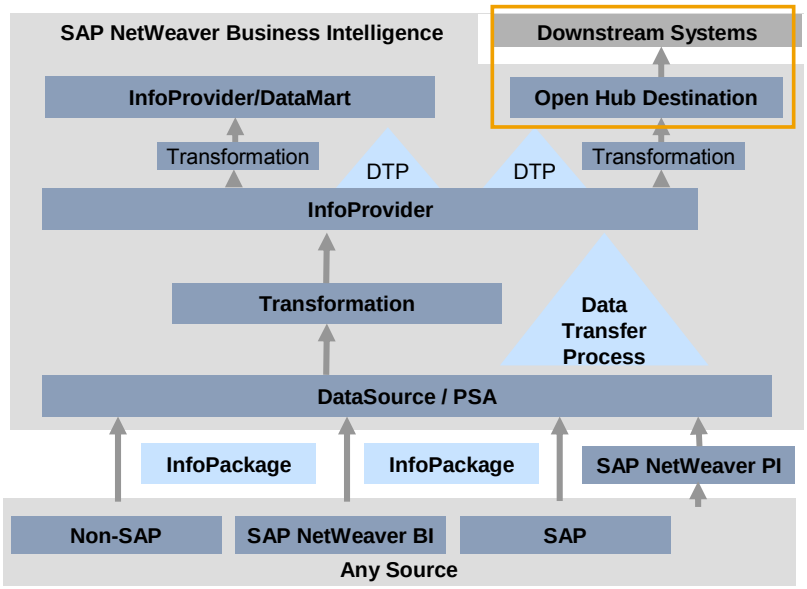
The open hub destination is the object that allows you to distribute data from a BI system to non-SAP data marts, analytical applications, and other applications. It allows you to ensure controlled distribution across multiple systems.

The open hub destination defines the target to which the data is relayed.

In earlier releases, the open hub destination was part of the InfoSpoke. It is now an independent object that provides more options as a result of its integration into the data flow.

The open hub service previously provided with the InfoSpoke can still be used. We recommend, however, that you use the new technology to define new objects.

The following figure outlines how the open hub destination is integrated into the data flow:



Use

Database tables (in the database for the BI system) and flat files can act as open hub destinations. You can extract the data from a database to non-SAP systems using APIs and a third-party tool.

Structure

The open hub destination contains all the information about a data target: the type of destination, the name of the flat file or database table and their properties, and the field lists and its properties.

Integration

You can use the data transfer process to update data to the open hub destination. This involves a transformation of the data. Not all rules types are available in the transformation for an open hub destination: Reading master data, time conversion, currency translation and unit conversion are not available.

4.1.6.8.1 Creating Open Hub Destinations

Procedure

1. In the *Modeling* area of the Data Warehousing Workbench, choose the open hub destination tree.
2. In the context menu of your InfoArea, choose *Create Open Hub Destination*.
3. Enter a technical name and a description. We recommend that you use the object from which you want to update data to the open hub destination as the template.

4. On the *Destination* tab page, select the required destination. The other settings you can make on this tab page differ depending on the destination you select. For more information, see
   - Database Tables As Destinations [Page 653]
   - Files As Destinations [Page 653]
   - Third-Party Tools As Destinations [Page 654]
5. On the *Field Definition* tab page, edit the field list. For more information, see Field Definitions [Page 656].
6. Activate the open hub destination.

Result

You can now use the open hub destination as a target in a data transfer process.

4.1.6.8.2 Database Tables As Destinations

Use

You can select a database table as an open hub destination.

Features

Generating Database Tables

When you activate the open hub destination, the system generates a database table. The generated database table has the prefix /BIC/OHxxx (xxx is the technical name of the destination).

Deleting Data from the Table

With an extraction to a database table, you can either retain the history of the data or just store the new data in the table. Choose *Delete Data from Table* when defining your destination if you want to overwrite the fields. In this case, the table is completely deleted and regenerated before each extraction takes place. We recommend that you use this mode if you do not want to store the history of the data in the table. If you do not select this option, the system only generates the table once before the first extraction. We recommend that you use this mode if you want to retain the history of the extracted data.

Note that if changes are made to the properties of the database table (for example, fields are added), the table is always deleted and regenerated.

Table Key Fields

You can choose whether you want to use a technical key or a semantic key.

Technical key:

If you set the *Technical Key* indicator, a unique key is added that consists of the technical fields OHREQUID (open hub request SID), DATAPAKID (data package ID), and RECORD (sequential number of a data record to be added to the table within a data package). These fields display the individual key fields for the table.

Using a technical key with a target table is particularly useful if you want to extract into a table that is not deleted before extraction. If an extracted record has the same key as a record that already exists, the duplicate records cause a short dump.

Semantic key:

If you set the *Semantic Key* indicator, the system selects all the fields in the field list as semantic keys, if they are suitable. You can change this selection in the field list. However, note that duplicate records may result from using a semantic key.

4.1.6.8.3 Files As Destinations

Use

You can select flat files in format *.CSV* as an open hub destination.

Features

The only file format that is supported for extraction to flat files is *.CSV.* A control file with information about the metadata is also generated. You can either save the file on the application server or in a local directory.

If you save the file locally, the file size must not exceed a half GB. When transferring mass data, you should save the file on the application server.

If the data is to be written to a BI system application server, you can determine the file name in two ways:

- File name:

  The file name is made up of the technical name of the open hub destination and the suffix *.CSV*. You cannot change this name.

- Logical file name:

  You can use input help to select a logical file name that you have already defined in Customizing. Create a logical path and assign a logical file name to it (see Defining Logical Path and File Names [External]).

  A logical file name can be made up of fixed path information, but also of variables such as calendar day and time. Logical file names can be transported.

If you save the file in a local directory, you cannot change the name of the file. It is made up of the technical name of the open hub destination and the suffix *.CSV*. The associated control file also has the prefix S_.

4.1.6.8.4 Third-Party Tools As Destinations

Use

You can use the open hub destination to extract data to non-SAP systems. Various APIs allow you to connect a third-party tool to the BI system and to use this third-party tool to distribute data to other non-SAP systems.

Features

First you extract the data from BI InfoProviders or DataSources into a database table in the BI system. The third-party tool receives a message when the extraction process is complete. You can define parameters for the third-party tool. You can also use the monitor to oversee the process.

You can connect one or more data transfer processes to an open hub destination of type *Third-Party Tool*.

You can use a process chain to start the extraction process not only in the BI system itself, but also using the third-party tool.

The following APIs are available:

RSB_API_OHS_DEST_SETPARAMS: You use this API to transfer the parameters of the third-party tool that are required for the extraction to the BI system. These parameters are saved in a parameter table within the BI system in the metadata for the open hub destination.

RSB_API_OHS_3RDPARTY_NOTIFY: This API sends a message to the third-party tool after extraction. It transfers the open hub destination, the request ID, the name of the database table, the number of extracted data records and the time stamp. In addition, you can add another parameter table that contains the parameters that are only relevant for the third-party tool.

RSB_API_OHS_REQUEST_SETSTATUS: This API sets the status of extraction to the third-party tool in the monitor. Red means that the existing table is not overwritten. If the status is green, the request is processed further. Normally the user can change the status manually in the monitor or in the maintenance screen for the data transfer process. However, these manual functions are deactivated with open hub destinations of type *Third-Party Tool*.

RSB_API_OHS_DEST_GETLIST: This API delivers a list of all open hub destinations.

RSB_API_OHS_DEST_GETDETAIL: This API gets the details of an open hub destination.

RSB_API_OHS_DEST_READ_DATA: This API reads data from the database table in the BI system.

For information on the parameters of the APIs, see:

API: RSB_API_OHS_DEST_SETPARAMS [External]

API: RSB_API_OHS_3RDPARTY_NOTIFY [External]

API: RSB_API_OHS_REQUEST_SETSTATUS [External]

API: RSB_API_OHS_DEST_GETLIST [External]

API: RSB_API_OHS_DEST_GETDETAIL [External]

API: RSB_API_OHS_DEST_READ_DATA [External]

Process Flow:

Extraction to the third-party tool can be executed as follows:

1. You define an open hub destination with *Third-Party Tool* as the destination type.
2. You create an RFC destination for your third-party tool and enter it in the definition of the open hub destination.
3. You use API RSB_API_OHS_DEST_SETPARAMS to define the parameters for the third-party tool that are required for the extraction.
4. You either start extraction immediately or include it in a process chain. You can also start this process chain from the third-party tool using process chain API

RSPC_API_CHAIN_START. The extraction process then writes the data to a database table in the BI system.

5. When the extraction process is finished, the system sends a notification to the third-party tool using API RSB_API_OHS_3RDPARTY_NOTIFY.
6. The extracted data is read by API RSB_API_OHS_DEST_READ_DATA.
7. The status of the extraction is transferred to the monitor by API RSB_API_OHS_REQUEST_SETSTATUS.

See also:

For detailed information about certification and the scenario, see the SDN at **www.sdn.sap.com** → *Partners and ISVs* → *SAP Integration and Certification Center* → *Integration Scenarios* → *Business Intelligence* → *Interface: BW-OHS*.

4.1.6.8.5 Field Definition

Use

On the *Field Definition* tab page you define the properties of the fields that you want to transfer.

Features

We recommend that you use a template as a basis when you create the open hub destination. The template should be the object from which you want to update the data. This ensures that all the fields of the template are available as fields for the open hub destination. You can edit the field list by removing or adding fields. You can also change the properties of these fields.

You have the following options for adding new fields:

- You enter field names and field properties, independent of a template.
- You select an InfoObject from the *Template InfoObject* column. The properties of the InfoObject are transferred into the rows.
- You choose *Select Template Fields*. A list of fields are available as fields for the open hub destination that are not contained in the current field list. You transfer a field to the field list by double-clicking on it. This allows you to transfer fields that had been deleted back into the field list.

If you want to define the properties of a field so that they are different from the properties of the template InfoObject, delete the template InfoObject entries for the corresponding field and change the properties of the field. If there is a reference to a template InfoObject, the field properties are always transferred from this InfoObject.

The file or database table that is generated from the open hub destination is made up of the fields and their properties and not the template InfoObjects of the fields.

If the template for the open hub destination is a DataSource, field SOURSYSTEM is automatically added to the field list with reference to InfoObject 0SOURSYSTEM. This field is required if data from heterogeneous source systems is being written to the same database table. The data transfer process inserts the source system ID that is relevant for the connected DataSource. You can delete this field if it is not needed.

If you have selected *Database Table* as the destination and *Semantic Key* as the property, the field list gets an additional column in which you can define the key fields for the semantic key.

In the *Format* column, you can specify whether you want to transfer the data in the internal or external format. For example, if you choose *External Format* here, leading zeros will be removed from a field that has an ALPHA conversion routine when the data is written to the file or database table.

4.1.6.9 Remodeling InfoProviders

Use

You want to modify an InfoCube into which data has already been loaded. You use remodeling to change the structure of the object without losing data.

If you want to change an InfoCube into which no data has yet been loaded, you can change it in InfoCube maintenance.

You may want to change an InfoProvider that has already been filled with data for the following reasons:

- You want to replace an InfoObject in an InfoProvider with another, similar InfoObject. You have created an InfoObject yourself but want to replace it with a BI Content InfoObject.
- The structure of your company has changed. The changes to your organization make different compounding of InfoObjects necessary.

Prerequisites

As a precaution, make a backup of your data before you start remodeling.

In addition, ensure that:

- You have stopped any process chains that run periodically and affect the corresponding InfoProvider. Do not restart these process chains until remodeling is finished.
- There is enough tablespace available in the database.
- After remodeling, you have to check which BI objects that are connected to the InfoProvider (for example, transformation rules, MultiProviders) have been deactivated. You have to reactivate these objects manually. The remodeling makes existing queries that are based on the InfoProvider invalid. You have to manually adjust these queries according to the remodeled InfoProvider. If, for example, you have deleted an InfoObject, you also have to delete it from the query.

Features

A remodeling rule is a collection of changes to your InfoCube that are executed simultaneously.

For InfoCubes, you have the following remodeling options:

For characteristics:

- Inserting or replacing characteristics with:
  - A constant

  - An attribute of an InfoObject within the same dimension
  - A value of another InfoObject within the same dimension
  - A customer exit (for user-specific code)
- Deleting characteristics

For key figures:

- Inserting:
  - A constant
  - A customer exit (for user-specific code)
- Replacing key figures with:
  - A customer exit (for user-specific code)
- Deleting key figures

You cannot replace or delete units. This avoids having key figures in the InfoCube without the corresponding unit.

SAP NetWeaver 2004s does not yet support the remodeling of InfoObjects or DataStore objects. This is planned for future releases.

Transport Connection

The remodeling is connected to the BI transport system. More information: Transport System [External]

4.1.6.9.1 Remodeling InfoProviders

Prerequisites

You have created an InfoProvider and loaded data into it.

We recommend that you compress the InfoCube if you want to add or replace key figures. You can only remodel the InfoCube in a noncompressed form if you are certain that there are no duplicate records in the InfoCube.

You add a key figure that is filled with a constant to a noncompressed InfoCube. The value of the key figure is valid for every row in the fact table of the InfoCube, including the rows that only differ in their request ID. During aggregation, the system also adds the values of the duplicates. You therefore get inconsistent values.

Procedure

1. To access InfoProvider remodeling in the Data Warehousing Workbench, choose *Administration.* You can also access it in the context menu of your InfoProvider in the InfoProvider tree by choosing *Additional Functions → Remodeling.*
2. Create a remodeling rule. Specify a name for the remodeling rule, select an InfoProvider as required, and choose *Create.*

3. Choose *Add Operation to List*. You can select one of the following options from the dialog box:
   - Add characteristic/key figure
   - Delete characteristic/key figure
   - Replace characteristic/key figure
4. For the *Insert Characteristic* and *Replace Characteristic* options, you have to specify how you want to fill the new characteristic with data:
   - Constant: the system fills the new characteristic with a constant.
   - Attribute: the system fills the new characteristic with the values of an attribute of a characteristic that is contained in this dimension.
   - 1:1 mapping for characteristic: the system fills the new characteristic with the values of another characteristic. For example, you replace one characteristic with another and, in doing so, adopt the value of the original characteristic.
   - Customer exit: the system fills the new characteristic using a customer exit. More information: Customer Exits in Remodeling [Page 231]

   For key figures, the customer exit is the only available fill method.
5. Choose *Transfer.*
6. Repeat steps 3 and 4 until you have collected all changes. In the bottom half of the screen, you can make changes to the operations at any point. To do this, select the corresponding step in the upper half of the screen.

   You can delete an operation at any time by choosing *Remove Operation from the List.*
7. Save your entries.
8. Choose *Check.* The system checks whether the operations are correctly defined and whether the relevant InfoObjects are available in the InfoProvider.
9. You can display a list of the objects affected by remodeling by choosing *Impact Analysis.* These objects are deactivated during remodeling. You can make a note of the objects that you need to reactivate later.
10. Choose *Schedule* either to start remodeling immediately or to schedule it for later. You can specify whether the process is to be executed in parallel by choosing the *Execute Steps in Parallel* indicator. Only set this indicator if your system has the capacity to support this.
11. You can check the process in the monitor. More information: Monitor and Error Handling [Page 231]

Result

The InfoProvider is available in a remodeled and active form.

4.1.6.9.2 Customer Exits in Remodeling

Use

In remodeling, you can use a customer exit to fill added or replaced characteristics and key figures with initial values.

Procedure

1. To implement a user-specific class, in the SAP Easy Access menu, choose *Tools* → *ABAP Workbench* → *Development* → *Class Builder*.
2. Enter a name for your implementation and choose create.
3. Select *Class* as the object type.
4. Enter a description and select the following options:
   - Instantiation: Public
   - Usual ABAP Class
   - Final

   Save your entries. The Class Builder appears.
5. Use the IF_RSCNV_EXIT interface. This has the EXIT method with the following parameters:

Parameter	Description
I_CNVTABNM	Name of the remodeled table. You require this parameter if you want to use the same customer exit for more than one remodeling rule.
I_R_OLD	Structure of the table before remodeling
C_R_NEWFIELD	Result of the routine, which is assigned to the new field

6. Implement your class and save your entries.

Result

All classes that implement the IF_RSCNV_EXIT interface are available in remodeling. They are listed in the input help for the customer exit.

4.1.6.9.3 Monitor and Error Handling

Use

You can monitor the remodeling using the monitor.

Features

The status of a request and the corresponding steps are displayed in the monitor.

When a remodeling is executed, errors may occur for several reasons:

- Problems with the database (insufficient tablespace, duplicate keys, partitioning, and so on)
- Problems with the application server caused by large volumes of data (timeout, and so on)
- Problems caused by conversion routines

If an error occurs, you can restart the request in the monitor. In the context menu for the request, you choose *Reset Request*. In the same menu, you then choose *Restart Request*.

In exceptional cases, it is only possible to reset a single step. In the context menu of one of the steps, you choose *Step: Reset.* The reset step is then executed again if the request is restarted.

4.1.7 Transformation

Use

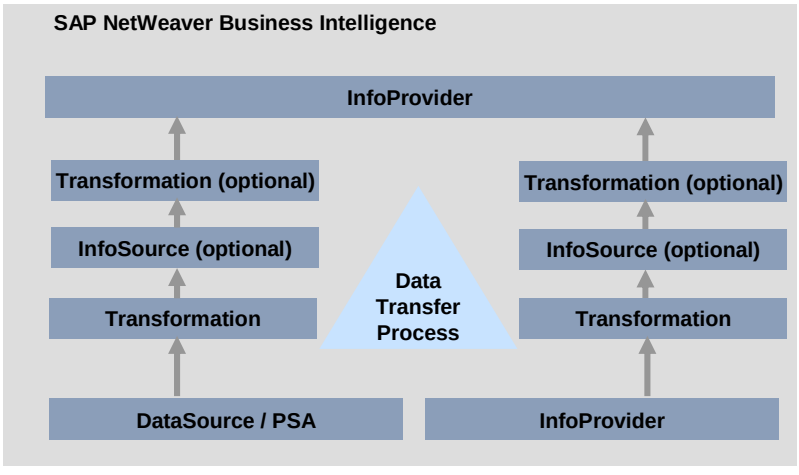
The transformation process allows you to consolidate, cleanse, and integrate data. You can semantically synchronize data from heterogeneous sources.

When you load data from one BI object into a further BI object, the data is passed through a transformation. A transformation converts the fields of the source into the format of the target.

Features

You create a transformation between a source and a target. The BI objects DataSource, InfoSource, DataStore object, InfoCube, InfoObject and InfoSet serve as source objects. The BI objects InfoSource, InfoObject, DataStore object and InfoCube serve as target objects.

The following figure illustrates how the transformation is integrated in the dataflow:



A transformation consists of at least one transformation rule. Various rule types, transformation types, and routine types are available. These allow you to create very simple to highly complex transformations:

- Transformation rules: Transformation rules map any number of source fields to at least one target field. You can use different rules types for this.
- Rule type: A rule type is a specific operation that is applied to the relevant fields using a transformation rule.

  For more information, see Rule Type [Page 233].
- Transformation type: The transformation type determines how data is written into the fields of the target.

  For more information, see Aggregation Type [Page 240].
- Rule group: A rule group is a group of transformation rules. Rule groups allow you to combine various rules.

  For more information, see Rule Group [Page 242].
- Routine: You use routines to implement complex transformation rules yourself. Routines are available as a rule type. There are also routine types that you can use to implement additional transformations.

  For more information, see Routines in the Transformation [Page 246].

4.1.7.1 Rule Type

Use

The rule type determines whether and how a characteristic or key figure, or a data field or key field is updated into the target.

Features

The following options are available:

Direct Assignment:

The field is filled directly from the selected source InfoObject. If the system does not propose a source InfoObject, you can assign a source InfoObject of the same type (amount, number, integer, quantity, float, time) or you can create a routine.

If you assign a source InfoObject to a target InfoObject that has the same type but a different currency, you have to translate the source currency into the target currency using a currency translation, or apply the source currency.

If you assign a source InfoObject to a target InfoObject that has the same type but a different unit of measure, you have to convert the source unit of measure into the target unit of measure using a unit of measure conversion, or apply the unit of measure from the source.

Constant:

The field is not filled by the InfoObject; it is filled directly with the value specified.

Formula:

The InfoObject is updated with a value determined using a formula.

For more information, see Transformation Library and Formula Builder [Page 236]

Read Master Data:

The InfoObject is updated by reading the master data table of a characteristic that is included in the source with a key and a value and that contains the corresponding InfoObject as an attribute. The attributes and their values are read using the key and are then returned.

The *Financial Management Area* characteristic is included in the target but does not exist in the source as a characteristic. However, the source contains a characteristic (cost center, for example) that has the *Financial Management Area* characteristic as an attribute. You can read the *Financial Management Area* attribute from the master data table and use it to fill the *Financial Management Area* characteristic in the target.

It is not possible to read recursively, that is, to read additional attributes for the attribute. To do this, you have to use routines.

If you have changed master data, you have to execute the change run. By reading the master data, the active version is read. If this is not available, an error occurs.

If the attribute is time dependent, you also have to define when it should be read: at the current date (sy-date), at the beginning or end of a period (defined by a time characteristic in the InfoSource), or at a constant date that you enter directly. Sy-date is used as the default.

Routine:

The field is filled by the transformation routine you have written.

The system allows you to select whether the routine is to be valid for all attributes for this characteristic, or only for the attributes displayed.

Transformation rules generally only have one return value. If you select *Return Table*, the corresponding key figure routine no longer has a return value; it has a return table. You can then generate any number of values from a data record.

For DataStore objects and InfoObjects: you cannot use the return code in the routine for data fields that are updated by being overwritten. If you do not want to update specific records, you can delete these from the start routine.

If, for the same characteristic, you generate different rules for different key figures or data fields, a separate data record can be created for each key figure from a data record of the source.

With InfoCubes: You can also select *Routine with Unit*. The return parameter 'UNIT' is then also added to the routine. You can store the required unit of the key figure, such as 'ST', in this parameter. You can use this option, for example, to convert the unit KG in the source, into tons in the target.

If you fill the target key figure from a transformation routine, currency translation has to be performed using the transformation routine. This means that automatic calculation is not possible.

Time Update:

When performing a time update, automatic time conversion and time distribution are available.

Direct Update: the system automatically performs a time conversion.

Time Conversion:

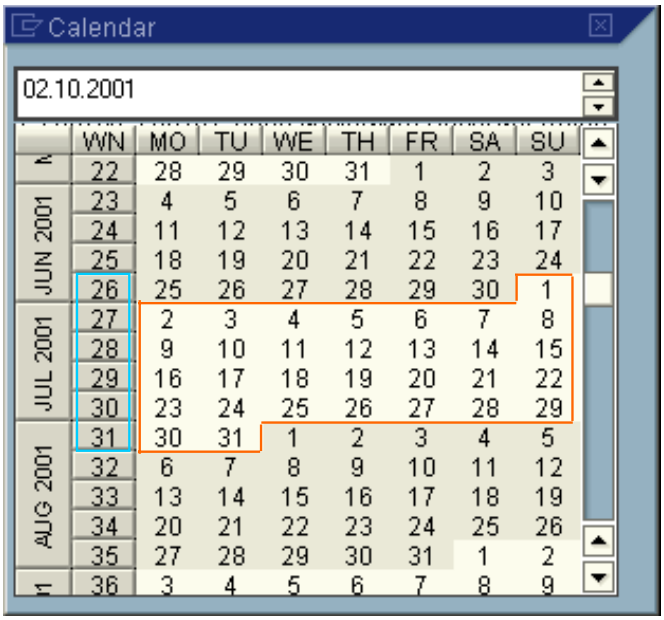
You can update source time characteristics to target time characteristics using automatic time conversion. This function is not available for DataStore objects, since time characteristics are treated as normal data fields. The system only displays the time characteristics for which an automatic time conversion routine exists.

Time Distribution:

You can update time characteristics with time distribution. All the key figures that can be added are split into correspondingly smaller units of time. If the source contains a time characteristic (such as 0CALMONTH) that is not as precise as a time characteristic of the target (such as 0CALWEEK), you can combine these characteristics with one another in the rule. The system then performs time distribution in the transformation.

For example, you break down the calendar month 07.2001 into the weeks 26.2001, 27.2001, 28.2001, 29.2001, 30.2001 and 31.2001. Each key figure that can be added receives 1/31 of the original value for week 26.2001, 7/31 for each of weeks 27, 28, 29, and 30, and exactly 2/31 of it for week 31.

The example is clearer if you compare it with the following calendar:



The time distribution is always applied to all key figures.

Initial:

The field is not filled. It remains empty.

No Transformation:

The key figures are not written to the InfoProvider.

Unit of Measure Conversion and Currency Translation

You can convert data records into the unit of measure or currency in the target transformation.

For more information, see:

- Currency Translation During Transformation [External]
- Quantity Conversion During the Transformation [External]

4.1.7.1.1 The Transformation Library and Formula Builder

Use

A transformation library is available in the maintenance for transformation rules (and in the update rules). You can use this in connection with the formula builder.

Do not use formulas for VirtualProviders because inversion is not allowed for them. Use routines in this case.

Features

The transformation library, in collaboration with the formula builder, enables you to easily create formulas, without using ABAP coding.

The transformation library has over 70 pre-defined functions, in the following categories:

- Functions for character strings
- Date functions
- Basic functions
- Mathematical functions
- Suitable functions
- Miscellaneous functions

In the dialog box to select an update method, you can use the information pushbutton to get a list of the available functions with a description of their syntax.

You also have the option to implement self-defined functions in the transformation library of the formula builder. You can integrate existing function modules in these self-defined functions. In doing so, you can also make available special functions to be used frequently that are not contained in the transformation library. Refer to BAdI: Customer-defined Functions in the Formula Builder [Page 237].

The formula builder has two modes: Standard and expert mode. In the standard mode, you can only enter the formulas using the pushbuttons and by double clicking on functions and fields. In the expert mode, however, you can enter formulas directly. You can also toggle between the tow modes when entering a formula.

You can find more detailed operating instructions for the formula builder by means of the information button .

You can find a step-by-step guide using an example under Example for Using the Formula Builder [Page 237].

4.1.7.1.1.1 Example for Using the Formula Editor

The company code field (0COMP_CODE) is not included in your data target or InfoSource. However you can determine the company code from the first four character spaces of the cost center (0COSTCENTER).

You create the following formula for this purpose:

```
SUBSTRING( cost center, '0' , '4')
```

Syntax:

```
SUBSTRING( String, Offset , Länge )
```

Step-by-Step Procedure in Standard mode:

1. In the transformation library, on the right hand side under *Show Me*, choose the category *Strings.* From the list, select the *Substring* function by double-clicking on it. The syntax of the formula is displayed in the formula window: `SUBSTRING( , , )`

   The cursor automatically appears over the first parameter that needs to be specified.
2. From the list on the left-hand side of the screen, choose the *Cost Center* field by double-clicking on it.
3. Place the cursor where you want to enter the next parameter.
4. Enter the number 0 using the *Constant* button (for the *Offset* parameter). The commas are added automatically.
5. Place the cursor where you want to enter the next parameter.
6. Enter the number 4 using the *Constant* button (for the *Length* parameter).
7. Choose *Back*. The formula is now checked and saved if it is correct. You receive a message if errors occurred during the check, and the system highlights the erroneous element in color.

4.1.7.1.1.2 BAdI: Customer-Defined Functions in the Formula Builder

Use

You can integrate self-defined functions in the transformation library of the formula builder. In doing so, you can also make special functions available for frequent use that are not contained in the transformation library. The Business Add-In RSAR_CONNECTOR is available for this. For more information about using Business Add-Ins (BAdIs), see Classic BAdIs [External].

Procedure

Implementing the BAdI

To implement a BAdI, see Implementing a Classic BAdI [External]. The specific special features for the implementation of the BAdI RSAR_CONNECTOR are described below.

1. Call transaction SE19. Enter RSAR_CONNECTOR as the name for the Add-In, for which you want to create the implementation.

By double-clicking the method (GET), the class builder appears. You can enter your coding for implementing the enhancement. For more information, see Structure of Implementation of a Function [Page 239]and Implementation of a Category [Page 239].

A function does not have a type. Therefore, the *TYPE* field of structure SFBE0PRND cannot be filled.

2. Save and activate your implementation.

Naming Conventions

The technical name of a self-defined function:

- cannot be empty
- must be consistent
- must begin with ‘C_’
- can only contain alphanumeric characters, that is, 'ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789_' (small letters, blank spaces and special characters are not allowed)
- can have a maximum of 61 characters

Implementing the Methods

The ABAP methods specified in the function description under class and method are called later in maintenance and formula evaluation. Therefore, the customer-defined functions must additionally be implemented as methods for BAdI implementation in an additional class. These methods must have the following properties:

- They are declared static and public.
- They can only have importing, exporting, and returning parameters. Changing parameters are not permitted.
- They can only have one exporting or returning parameter.
- Exporting parameters cannot have a generic type.

In the methods, you can use ABAP code to implement the function.

The system does not check whether the class or method specified in BAdI implementation actually exists. If a class or method does not exist, a runtime error occurs when the function is used in the formula builder.

Result

The functions you have defined are available in the transformation library in the *Customer-Defined Functions* selection.

4.1.7.1.1.2.1 Structure of Implementation of a Function

The following table explains the structure of implementation of a function:

Coding Lines	Description
`method IF_EX_RSAR_CONNECTOR~GET .`	
`data: l_function type SFBEOPRND.`	Structure with the description of the function
`case i_key.`	Importing parameter: key with the function category
`when 'CUSTOM'.`	The BAdI implementation is always accessed with the 'CUSTOM' key.
`* description of function C_TECH_NAME 1`	
`clear l_function.`	
`l_function-tech_name = 'C_TECH_NAME1'.`	Appears later in the *Technical Name* column and must be unique.
`l_function-descriptn = 'description 1'.`	Appears later in the *Description* column.
`l_function-class = 'CL_CUSTOM_FUNCTIONS'.`	Name of the class in which the function is implemented.
`l_function-method = 'CUSTOMER_FUNCTION1'.`	Name of the method in which the function is implemented.
`APPEND l_function TO c_operands.`	Changing parameter: table with descriptions of the function
`* ... further descriptions`	
`endcase.`	
`endmethod.`	

4.1.7.1.1.2.2 Implementation of a Category

You can implement your own categories and group your own functions under them.

Add the following code in the BAdI implementation:

Coding Lines	Description
`data: l_s_operand TYPE SFBEOPRND.`	
`if i_key = SPACE.`	
`l_s_operand-descriptn = <description>.`	Description of category
`l_s_operand-tech_name = <name>.`	Name of category in uppercase letters
`APPEND l_function TO c_operands.`	

exit.	
endif.	

To group functions in this category, add the following code to the BAdI implementation:

Coding Lines	Description
if i_key = <name of group>.	
l_s_operand-descriptn = <description>.	Description of function
l_s_operand-tech_name = <name>.	Name of function in uppercase letters
l_s_operand-tech_name = <name>.	Name of class that implements BAdI
l_s_operand-tech_name = <name>.	Name of method that implements BAdI
APPEND l_function TO c_operands.	
exit.	
endif.	

4.1.7.2 Aggregation Type

Use

You use the aggregation type to control how a key figure or data field is updated to the InfoProvider.

Features

For InfoCubes:

Depending on the aggregation type you specified in key figure maintenance for this key figure, you have the options *Summation,* or *Maximum* or *Minimum*. If you choose one of these options, new values are updated to the InfoCube.

The aggregation type (summation, minimum & maximum) specifies how key figures are updated if the primary keys are the same. For new values, either the total, the minimum, or the maximum for these values is formed.

For InfoObjects:

Only the *Overwrite* option is available. With this option, new values are updated to the InfoObject.

For DataStore Objects:

Depending on the type of data and the DataSource, you have the options *Summation, Minimum, Maximum* or *Overwrite.* When you choose one of these options, new values are updated to the DataStore object.

For numerical data fields, the system uses characteristic 0RECORDMODE to propose an update type. If only the after-image is delivered, the system proposes *Overwrite*. However, it may be useful to change this: For example, the counter data field “# Changes” is filled with a constant 1, but still has to be updated (using addition), even though an after-image only is delivered.

The characteristic 0RECORDMODE is used to pass DataSource indicators (from SAP systems) to the update.

If you are not loading delta requests to the DataStore object, or are only loading from file DataSources, you do not need the characteristic 0RECORDMODE.

Summation:

Summation is possible if the DataSource is enabled for an additive delta. Summation is not supported for data types CHAR, DAT, TIMS, CUKY or UNIT.

Overwrite:

Overwrite is possible if the DataSource is delta enabled.

When the system updates data, it does so in the chronological order of the data packages and requests. It is your responsibility to ensure the logical order of the update. This means, for example, that orders must be requested before deliveries, otherwise incorrect results may be produced when you overwrite the data. When you update, requests have to be serialized.

Example

You are loading data to a DataStore object. In this example, the order quantity changes after the data is loaded into the BI system. With the second load process, the data is overwritten because it has the same primary key.

First Load Process

Document No.	Document Item	Order Quantity	Unit of Measure
100001	10	200	Pieces
100001	20	150	Pieces
100002	10	250	kg

Second Load Process

Document No.	Document Item	Order Quantity	Unit of Measure
100001	10	180	Pieces
100001	20	165	Pieces

4.1.7.3 Rule Group

Use

A rule group is a group of transformation rules. It contains one transformation rule for each key field of the target. A transformation can contain multiple rule groups.

Rule groups allow you to combine various rules. This means that for a characteristic, you can create different rules for different key figures.

Features

Each transformation initially contains a standard group. Besides this standard group, you can create additional rule groups.

If you have defined a new rule in rule details, you can specify whether this rule is to be used as a reference rule for other rule groups. If it is used as a reference rule, then this rule is also used in existing rule groups as a reference rule where no other rule has been defined.

Example

The source contains three date characteristics:

- Order date
- Delivery date
- Invoice date

The target only contains one general date characteristic. Depending on the key figure, this is filled from the different date characteristics in the source.

Create three rule groups which, depending on the key figure, update the order date, delivery date, or invoice date to the target.

4.1.7.4 Creating Transformations

Procedure

You are in the *Modeling* functional area of the Data Warehousing Workbench.

1. In the InfoProvider tree, choose *Create Transformation* in the context menu for your InfoProvider.
2. Select a source for your transformation and choose *Create Transformation*.
3. The system proposes a transformation. You can use this transformation as it is, or modify it to suit your requirements. To the left of the screen, you see the source; to the right of the screen, you see the rule group. To show the target as well, choose *Switch Detail View On/Off.*

   For InfoCubes with non-cumulative key figures, you cannot change the transformation suggested by the system. These transformation rules fill the time reference characteristic of the InfoCube. All other time characteristics are automatically derived from the time-reference characteristic.

4. Using the mouse, you can create new connecting arrows or change existing connecting arrows; you can delete a connecting arrow by choosing the relevant entry in the context menu for the arrow.
5. You can activate the check for referential integrity in the rule group for single characteristics.

   The check for referential integrity determines the valid values of the characteristic. It is checked against the master data table (meaning the SID table) or against a DataStore object that is specified in the InfoObject maintenance for this characteristic.
6. If you double-click an InfoObject in the transformation group, the maintenance screen for the rule details is displayed. You can:
   - Select a rule type.

     More information: Rule Type [Page 233]
   - Activate the conversion routine, if it exists. It is deactivated by default because the system assumes that the DataSource provides the internal format.

     More information: Conversion Routines in BI Systems [Page 98].
   - With key figures, you can specify a transformation type and define a currency translation or quantity conversion. More information:
     - Currency Translation During Transformation [External]
     - Quantity Conversion During the Transformation [External]
   - Using the *InfoObject Assignment* field for a source field, you can assign an InfoObject to a DataSource from which the data is to be read. This is required to read master data and for currency translations and quantity conversions. More information:
     - Assigning InfoObjects for Reading Master Data [Page 244]
     - Assigning InfoObjects for Converting Amounts or Currencies [Page 245]
     - Assigning InfoObjects for Time Conversion [Page 245]
7. To create additional rule groups, choose *Rule Group* → *New Rule Group.*
8. To create the corresponding routines for your transformation, choose *Start Routine* and *End Routine*.

   See also: Routines in Transformations [Page 246]
9. Activate your transformation.

Result

The transformation is executed with the corresponding data transfer process when the data is loaded.

You can first simulate the transformation if you would like to check whether it provides the required results. To do so, execute the simulation of the DTP request. The transformation is also simulated along with the simulation of the data update.

More information: Simulating and Debugging DTP Requests [Page 335].

4.1.7.4.1 Assigning InfoObjects for Reading Master Data

Use

An InfoObject has to be assigned to a source field of a DataSource if master data needs to be read.

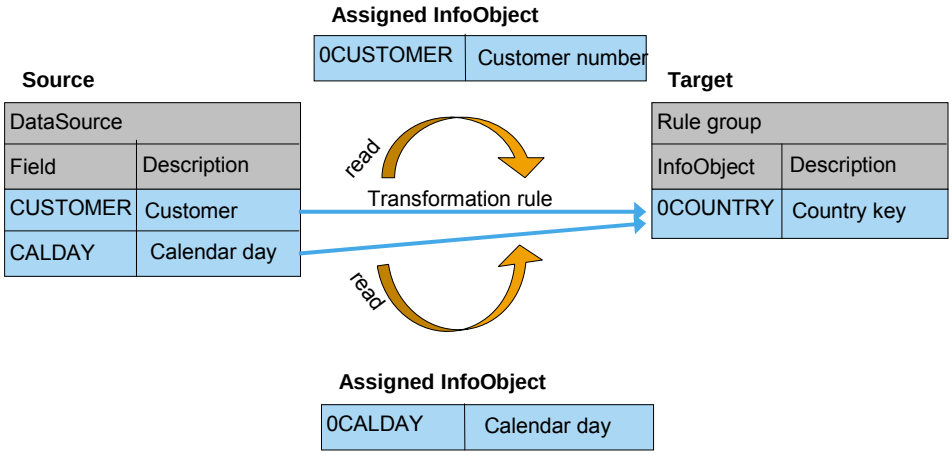
Procedure

1. You are in the rule details. Select the rule type *Time Update* or *Read Master Data.*
2. Select an InfoObject in the *InfoObject Assignment* field in the *Source Fields of Rule* area.
3. If the InfoObject is time-dependent, you usually have to add a time characteristic before you can specify the period. This must be an SAP time characteristic (0CALDAY, 0CALWEEK, 0CALMONTH, 0CALQUARTER, 0CALYEAR, 0FISCYEAR, 0FISCPER). To do so, choose *Add Source Fields* ( ) in the *Source Fields of Rule* area and then select a time characteristic.
4. Determine the time at which the master data needs to be read - on the current date (sy-date), on a constant date that you enter directly, or at the beginning or end of a period (determined by the time characteristic). To do so, click the *key date determination* icon ( ).
5. Choose *Transfer Values*.

Example

In your transformation, you want to assign the CUSTOMER field of the DataSource to the 0COUNTRY InfoObject in your target. To do so, you assign the 0CUSTOMER InfoObject to the CUSTOMER field in the rule details so that the properties of the InfoObject can be read.

To be able to specify time dependence for reading master data, assign the CALDAY field to the 0COUNTRY rule as an additional input field. The CALDAY field of the DataSource also needs an assigned InfoObject. Assign 0CALDAY to it so that 0CALDAY's properties can be read. Then enter a time.



4.1.7.4.2 Assigning InfoObjects for Converting Amounts or Currencies

Use

An InfoObject has to be assigned to a source field of a DataSource if currencies or units of measure need to be converted. The translation types require InfoObjects as input fields to carry out a conversion. You therefore have to assign the appropriate InfoObject to the source field.

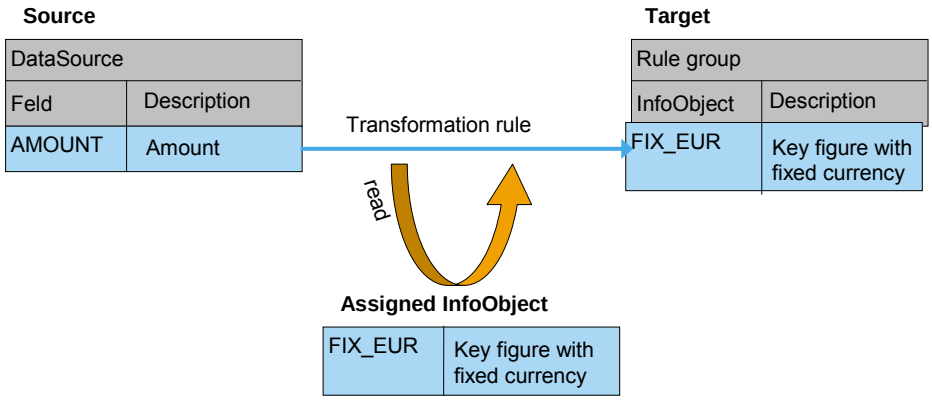
If the target has a fixed unit, you only have to assign an InfoObject if you want to carry out a conversion. Otherwise, you can select the *No Conversion* option.

Procedure

1. You are in the rule details. Select the rule type *Direct Assignment.*
2. Select an appropriate key figure in the *InfoObject Assignment* field in the *Source Fields of Rule* area.
3. Choose the required conversion in the *Currency* field.
4. Choose *Transfer Values*.

Example

In your transformation, you want to assign the AMOUNT field of the DataSource to the FIX_EUR InfoObject in your target, carrying out a currency conversion. To do so, you assign the FIX_EUR InfoObject to the AMOUNT field in the rule details so that the currency can be read from the InfoObject.



4.1.7.4.3 Assigning InfoObjects for Time Conversion

Use

An InfoObject has to be assigned to a source field of a DataSource if time conversion is required. It is required if the granularity of the source field is different from the granularity of the target field. Assigning a time characteristic allows the properties of the time characteristic to be adopted.

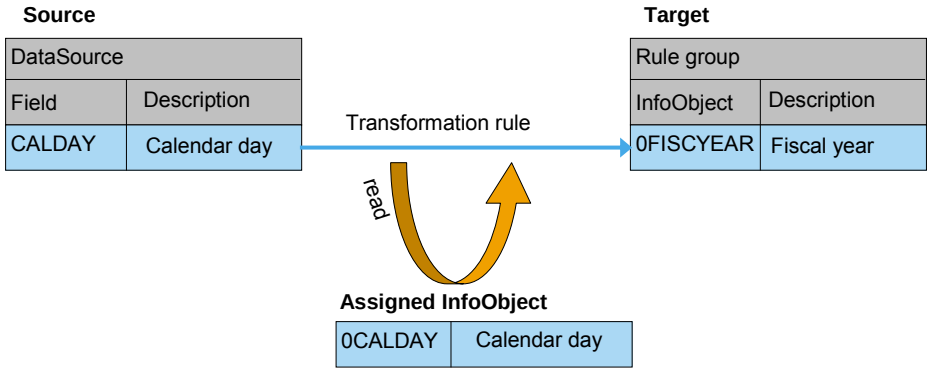
If you do not assign a time characteristic, a direct update takes place. The value is assigned directly and is truncated if necessary. Source fields of the DDIC type *DATS* are an exception. The system handles these fields as if a time characteristic were assigned.

Procedure

1. You are in the rule details. Select the rule type *Time Update.*
2. Select a time characteristic in the *InfoObject Assignment* field in the *Source Fields of Rule* area.
3. Choose *Transfer Values*.

Example

In your transformation, you want to assign the CALDAY field of the DataSource to the 0FISCYEAR InfoObject in your target. To do so, you assign the 0CALDAY InfoObject to the CALDAY field in the rule details so that the properties of the InfoObject can be read.



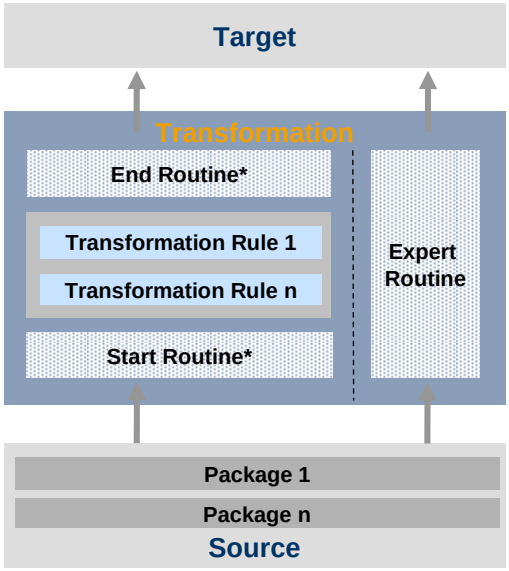
4.1.7.5 Routines in Transformations

Use

You use routines to define complex transformation rules.

Routines are local ABAP classes that consist of a predefined definition area and an implementation area. The TYPES for the inbound and outbound parameters and the signature of the routine (ABAP method) are stored in the definition area. The actual routine is created in the implementation area. ABAP object statements are available in the coding of the routine. Upon generation, the coding is embedded in the local class of the transformation program as the method.

The following graphic shows the position of these routines in the data flow:



* Optional

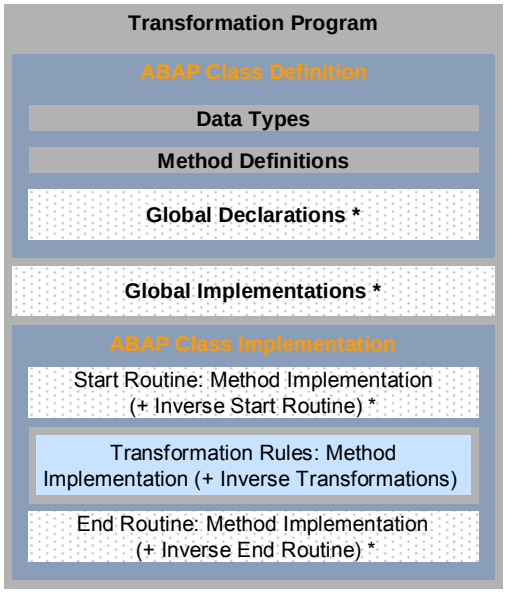
Features

The routine has a global part and a local part. In the global part you define global data declarations 'CLASS DATA'. These are available in all routines.

You can create function modules, methods or external subprograms in the ABAP Workbench if you want to reuse source code in routines. You can call these in the local part of the routine. If you want to transport a routine that includes calls of this type, the routine and the object called should be included in the same transport request.

Transformations include different types of routine: Start routines, routines for key figures or characteristics, end routines and expert routines.

The following figure shows the structure of the transformation program with transformation rules, start routine and end routine:



* optional

The following figure shows the structure of the transformation program with expert routine:

Start Routine

The start routine is run for each data package at the start of the transformation. The start routine has a table in the format of the source structure as input and output parameters. It is used to perform preliminary calculations and store these in a global data structure or in a table. This structure or table can be accessed from other routines. You can modify or delete data in the data package.

Routine for Key Figures or Characteristics

This routine is available as a rule type; you can define the routine as a transformation rule for a key figure or a characteristic. The input and output values depend on the selected field in the transformation rule. More information: the **Routine** section under Rule Type [Page 233].

End Routine

An end routine is a routine with a table in the target structure format as input and output parameters. You can use an end routine to postprocess data after transformation on a package-by-package basis. For example, you can delete records that are not to be updated, or perform data checks.

If the target of the transformation is a DataStore object, key figures are updated by default with the aggregation behavior **Overwrite** (MOVE). You have to use a dummy rule to override this.

Expert Routine

This type of routine is only intended for use in special cases. You can use the expert routine if there are not sufficient functions to perform a transformation. The expert routine should be used as an interim solution until the necessary functions are available in the standard routine.

You can use this to program the transformation yourself without using the available rule types. You must implement the message transfer to the monitor yourself.

If you have already created transformation rules, the system deletes them once you have created an expert routine.

If the target of the transformation is a DataStore object, key figures are updated by default with the aggregation behavior **Overwrite** (MOVE).

More Information:

Example: Start Routine [Page 250]

Example: Characteristic Routines [Page 251]

Example: End Routine [Page 252]

4.1.7.5.1 Creating Routines

Prerequisites

If you previously worked with the ABAP form routines of the update rules and transfer rules, you have to familiarize yourself with the differences in working with routines in the transformation.

See Differences in Routine Concepts [Page 265].

Procedure

You are in the routine editor. To create a routine, enter the following:

1. Between *$*$ begin of global ... and *$*$ end of global ... you can define the global data declarations 'CLASS DATA'. These are available in all routines. Data declarations with ‘DATA’ can only be accessed in the current package.

   This means that you can use intermediate results in other routines, for example, or reuse results when you call a routine again at a later time.

   When you perform serial loads, one process instance is used for the entire request. In this case, data with the ‘CLASS DATA’ data declaration can be accessed for the entire request (all packages).

   Several process instances are used when you perform parallel loads.

   A single process instance can be used a more than once, depending on the number of data packages to be processed and the number of available process instances. This means that with parallel loads, data with the 'CLASS DATA' data declaration is not initialized for each data package and may still contain data from predecessor packages.

   For this reason, use 'CLASS DATA' or 'DATA' for the global data, depending on the scenario.

   In the routine editor, a maximum of 72 characters per line are currently permitted. Any additional characters are cut off when you save.

2. Enter your program code for the routine between *$*$ begin of routine ... and *$*$ end of routine... . For information about the parameters of the routine, see

   - Start Routine Parameters [Page 252]

- Routine Parameters for Key Figures or Characteristics [Page 253]
- End Routine Parameters [Page 254]

Do not use a SAP COMMIT (ABAP statement: COMMIT WORK) in your coding. When this statement is executed, the cursor that is used from the source for reading is lost. Use a DB COMMIT (call function module DB_COMMIT) instead or avoid using such COMMITs altogether.

3. Check the syntax of your routine.
4. Save the routine. You end the maintenance session for the routine by leaving the editor.

More Information:

Example: Start Routine [Page 250]

Example: Characteristic Routines [Page 251]

Example: End Routine [Page 252]

4.1.7.5.1.1 Example: Start Routine

In the SAP ERP system, you are loading data using the *General Ledger: Transaction Figures* DataSource (FI_GL_1) into the DataStore object *FIGL: Transaction Figures* (0FIGL_O06).

You want to create a start routine that deletes all the records from a data package that have debit and credit postings that are equal to zero.

1. Create a transformation. The source of the transformation has the *Total Debit Postings* (UMSOL) and *Total Credit Postings* (UMHAB) fields. They are assigned to the InfoObjects *Total Debit Postings* (0DEBIT) and *Total Credit Postings* (0CREDIT).
2. Choose *Create Start Routine.* The routine editor opens.
3. You go to the local part of the routine. You enter the following lines of code:

```
*---------------------------------------------------------------------*
 METHOD start_routine.
*=== Segments ===

    FIELD-SYMBOLS:
      <SOURCE_FIELDS>    TYPE _ty_s_SC_1.

*$*$ begin of routine - insert your code only below this line        *-*

  DELETE SOURCE_PACKAGE where UMHAB = 0 and UMSOL = 0

*$*$ end of routine - insert your code only before this line         *-*
  ENDMETHOD.                    "start_routine
*---------------------------------------------------------------------*
```

The delete statement is the only line you require in order to filter debit and credit postings without values out of the data package.

4. You exit the routine editor.
5. You save the transformation. An edit icon next to the *Start Routine* indicates that a start routine is available.

4.1.7.5.1.2 Example: Characteristic Routine

In the SAP ERP system, you are loading data using the *General Ledger: Transaction Figures* DataSource (0FI_GL_1) into the DataStore object *FIGL: Transaction Figures* (0FIGL_O06).

You want to create a routine for the characteristic *Debit/Credit Indicator* (0FI_DBCRIND) in the target that assigns the value **D** to debit postings and the value **C** to credit postings.

6. You are in transformation maintenance. In the rule group, you double click on InfoObject *Debit/Credit Indicator* (0FI_DBCRIND). The rule details screen appears.
7. You choose *Add Source Fields* and add the *Total Debit Postings* (UMSOL) and *Total Credit Postings* (UMHAB) fields so that they are available in the routine.
8. You choose *Routine* as the rule type. The routine editor opens.
9. You enter the following lines of code. They return either **D** or a **C** as the result value:

```
*---------------------------------------------------------------------*
 METHOD compute_0FI_DBCRIND.

  DATA:
   MONITOR_REC    TYPE rsmonitor.

*$*$ begin of routine - insert your code only below this line        *-*

*    result value of the routine
    if SOURCE_FIELDS-umhab ne 0 and SOURCE_FIELDS-umsol eq 0.
      RESULT = 'D'.
    elseif SOURCE_FIELDS-umhab eq 0 and SOURCE_FIELDS-umsol ne 0.
      RESULT = 'C'.
    else.
      monitor_rec-msgid = 'ZMESSAGE'.
      monitor_rec-msgty = 'E'.
      monitor_rec-msgno = '001'.
      monitor_rec-msgv1 = 'ERROR, D/C Indicator'.
      monitor_rec-msgv2 = SOURCE_FIELDS-umhab.
      monitor_rec-msgv3 = SOURCE_FIELDS-umsol.
      append monitor_rec to monitor.
      RAISE EXCEPTION TYPE CX_RSROUT_ABORT.
    endif.

*$*$ end of routine - insert your code only before this line         *-*
  ENDMETHOD.                    "compute_0FI_DBCRIND
*---------------------------------------------------------------------*
```

   The system checks if the debit and credit postings contain values:

   - If the debit posting has values that are not equal to zero and the credit posting is equal to zero, the system assigns the value **D**.
   - If the credit posting has values that are not equal to zero and the debit posting is equal to zero, the system assigns the value **C**.
   - If both the debit and credit postings contain values, the system outputs an error in the monitor and terminates the loading process.

10. You exit the routine editor.
11. In the *Rule Details* dialog box, you choose *Transfer Values*.
12. You save the transformation.

4.1.7.5.1.3 Example: End Routine

In the SAP ERP system, you are loading data using the *General Ledger: Transaction Figures* DataSource (0FI_GL_1) into the DataStore object *FIGL: Transaction Figures* (0FIGL_O06).

You want to create an end routine to fill the additional InfoObject *Plan/Actual Indicator* (ZPLACTUAL). You also want the routine to read field *Value Type*. If the value is 10 (actual), value A is written to the *Plan/Actual Indicator* InfoObject; if the value is 20 (plan), value P is written to the *Plan/Actual Indicator* InfoObject.

1. You are in transformation maintenance. Choose *Create End Routine.* The routine editor opens.
2. You enter the following lines of code:

```
*---------------------------------------------------------------------*
 METHOD end_routine.
*=== Segments ===

    FIELD-SYMBOLS:
      <RESULT_FIELDS>    TYPE _ty_s_TG_1.

*$*$ begin of routine - insert your code only below this line        *-*

    loop at RESULT_PACKAGE assigning <RESULT_FIELDS>
      where vtype eq '010' or vtype eq '020'.
   case <RESULT_FIELDS>-vtype.
    when '010'.
     <RESULT_FIELDS>-/bic/zplactual = 'A'. "Actual
    when '020'.
     <RESULT_FIELDS>-/bic/zplactual = 'P'. "Plan
   endcase.
  endloop.
*$*$ end of routine - insert your code only before this line        *-*
 ENDMETHOD.                "end_routine
*---------------------------------------------------------------------*
```

   The code loops through result_package searching for values that have the value type 10 or 20. For these values, the appropriate value is passed on to InfoObject *Plan/Actual Indicator* (ZPLACTUAL).

3. You exit the routine editor.
4. You save the transformation. An edit icon next to the *End Routine* indicates that an end routine is available.

4.1.7.5.2 Start Routine Parameters

Importing

- REQUEST: Request ID
- DATAPAKID: Number of current data package

Exporting

- MONITOR: Table for user-defined monitoring. This table is filled by means of row structure MONITOR_REC (the record number of the processed record is inserted automatically from the framework).

Changing

- SOURCE_PACKAGE: Structure that contains the inbound fields of the routine.

Raising

- CX_RSROUT_ABORT: If a raise exception type cx rsrout_abort is triggered in the routine, the system terminates the entire load process. The request is highlighted in the extraction monitor as having been terminated. The system stops processing the current data package. This can be useful with serious errors.

4.1.7.5.3 Routine Parameters for Key Figures or Characteristics

Importing

- REQUEST: Request ID
- DATAPAKID: Number of current data package
- SOURCE_FIELDS: Structure with the routine source fields defined on the UI

Exporting

- MONITOR: Table for user-defined monitoring. This table is filled using row structure MONITOR_REC (the record number of the processed record is inserted automatically from the framework).
- RESULT: You have to assign the result of the computed key figure or computed characteristic to the RESULT variables.
- CURRENCY (optional): If the routine has a currency, you have to assign the currency here.
- UNIT (optional): If the routine has a unit, you have to assign the unit here.

Raising

Exception handling using exception classes is used to control what is written to the target:

- CX_RSROUT_SKIP_RECORD: If a `raise exception type cx_rsrout_skip_record` is triggered in the routine, the system stops processing the current row and continues with the next data record.
- CX_RSROUT_SKIP_VAL: If an `exception type cx_rsrout_skip_val` is triggered in the routine, the target field is deleted.
- CX_RSROUT_ABORT: If a `raise exception type cx rsrout_abort` is triggered in the routine, the system terminates the entire load process. The request is highlighted in the extraction monitor as *Terminated*. The system stops processing the current data package. This can be useful with serious errors.

4.1.7.5.4 End Routine Parameters

Importing

- REQUEST: Request ID
- DATAPAKID: Number of current data package

Exporting

- MONITOR: Table for user-defined monitoring. This table is filled using row structure MONITOR_REC (the record number of the processed record is inserted automatically from the framework).

Changing

- RESULT_PACKAGE: Contains all data that has been processed by the transformation.

Raising

- CX_RSROUT_ABORT: If a raise exception type cx rsrout_abort is triggered in the routine, the system terminates the entire load process. The request is highlighted in the extraction monitor as *Terminated.* The system stops processing the current data package. This can be useful with serious errors.

  Only fields that have a rule in the transformation are transferred from the end routine.

4.1.7.5.5 Creating Inversion Routines

Use

If you have defined routines in the transformation for a VirtualProvider, for performance reasons it may be useful to create inversion routines for these routines. In this way you can transform the selection criteria of a navigation step into selection criteria for the extractor. However, you do not require inversion routines to ensure the consistency of the data.

More information: Processing Selection Conditions [Page 169]

When you jump to a transaction in another SAP system using the report-report interface, you have to create an inversion routine for the transformation if you are using one, because otherwise the selections cannot be transferred to the source system.

You can create an inversion routine for all types of routine. The following rules apply:

- With expert routines, there is no segmentation into conditions.
- With start routines, the system performs segmentation into conditions. The system applies this to the complete source structure. The source structure is the start and end point.
- With end routines, the target structure is the start and end point.

Prerequisites

You have already created a routine.

Procedure

You are in the routine editor. To create an inversion routine, enter the following:

1. Between **$*$ begin of inverse routine ...* and **$*$ end of inverse routine ...* enter your program code to invert the routine.

   With an inversion routine for a VirtualProvider, it is sufficient if the value set is restricted in part. You do not need to specify an exact selection. The more exactly you restrict the selection, the better the system performance when you execute a query.

   With an inversion routine for a jump using the report-report interface, you have to make an exact inversion so that the selections can be transferred exactly.

   More information about the parameters of the routine: Parameters of Inversion Routines [Page 255]

2. Check the syntax of your routine.
3. Save the routine. You end the maintenance session for the routine by leaving the editor.

Example

An example for an inversion routine: Example for Inversion Routine [Page 256]

4.1.7.5.5.1 Inversion Routine Parameters

The inversion routine has method *invert*.

It has the following parameters:

Importing

- i_th_fields_outbound: Fields/InfoObjects for the query structure
- i_r_selset_outbound: Query selection conditions
- i_is_main_selection: Allows you to transfer complex selection conditions such as selection conditions for columns.
- i_r_selset_outbound_complete: All selections
- i_r_universe_inbound: Description of source structure with regard to set objects.

Changing

- c_th_fields_inbound: Fields/InfoObjects for the target structure
- c_r_selset_inbound: Taget selection conditions. You can fill the target field from more than one source field. In this case, you have to define more than one condition.
- c_exact: Allows you to specify whether you want the transformation of the selection criteria to be performed exactly. If the condition can be filled exactly, a direct call is possible. This is important when you call the report-report interface. If the condition cannot be filled exactly, a selection screen appears for the user.

4.1.7.5.5.2 Example for Inversion Routine

In this example, the German keys 'HERR' and 'FRAU' in the target characteristic are mapped to the English keys 'MR' and 'MRS' from the field PASSFORM (form of address) of the source. All other values from the source field are mapped to the initial value.

The coding of the routine is as follows:

```
*---------------------------------------------------------------------
-*
*       CLASS routine DEFINITION
*---------------------------------------------------------------------
-*
*
*---------------------------------------------------------------------
-*
CLASS lcl_transform DEFINITION.
  PUBLIC SECTION.

    TYPES:
      BEGIN OF _ty_s_SC_1,
*      Field: PASSFORM Anrede.
        PASSFORM           TYPE C LENGTH 15,
      END   OF _ty_s_SC_1.
    TYPES:
      BEGIN OF _ty_s_TG_1,
*      InfoObject: 0PASSFORM Anrede.
        PASSFORM           TYPE /BI0/OIPASSFORM,
      END   OF _ty_s_TG_1.
  PRIVATE SECTION.

    TYPE-POOLS: rsd, rstr.

*$*$ begin of global -
 insert your declaration only below this line  *-*
    DATA p_r_set_mr    TYPE REF TO cl_rsmds_set.
    DATA p_r_set_mrs   TYPE REF TO cl_rsmds_set.
    DATA p_r_set_space TYPE REF TO cl_rsmds_set.

*$*$ end of global -
 insert your declaration only before this line   *-*
    METHODS
      compute_0PASSFORM
        IMPORTING
          request                  type rsrequest
          datapackid               type rsdatapid
          SOURCE_FIELDS              type _ty_s_SC_1
        EXPORTING
          RESULT                   type _ty_s_TG_1-PASSFORM
          monitor                  type rstr_ty_t_monitor
        RAISING
          cx_rsrout_abort
          cx_rsrout_skip_record
          cx_rsrout_skip_val.
    METHODS
```

```
      invert_0PASSFORM
        IMPORTING
          i_th_fields_outbound         TYPE rstran_t_field_inv
          i_r_selset_outbound          TYPE REF TO cl_rsmds_set
          i_is_main_selection          TYPE rs_bool
          i_r_selset_outbound_complete TYPE REF TO cl_rsmds_set
          i_r_universe_inbound         TYPE REF TO cl_rsmds_universe
        CHANGING
          c_th_fields_inbound          TYPE rstran_t_field_inv
          c_r_selset_inbound           TYPE REF TO cl_rsmds_set
          c_exact                      TYPE rs_bool.
ENDCLASS.                    "routine DEFINITION
*$*$ begin of 2nd part global -
 insert your code only below this line  *
... "insert your code here
*$*$ end of 2nd part global -
 insert your code only before this line   *

*---------------------------------------------------------------------
-*
*       CLASS routine IMPLEMENTATION
*---------------------------------------------------------------------
-*
*
*---------------------------------------------------------------------
-*
CLASS lcl_transform IMPLEMENTATION.

  METHOD compute_0PASSFORM.

*   IMPORTING
*     request     type rsrequest
*     datapackid  type rsdatapid
*     SOURCE_FIELDS-PASSFORM TYPE C LENGTH 000015
*    EXPORTING
*      RESULT type _ty_s_TG_1-PASSFORM

    DATA:
      MONITOR_REC    TYPE rsmonitor.

*$*$ begin of routine -
 insert your code only below this line        *-*

    CASE SOURCE_FIELDS-passform.
      WHEN 'HERR'. RESULT = 'MR'.
      WHEN 'FRAU'. RESULT = 'MRS'.
      WHEN OTHERS. RESULT = space.
    ENDCASE.

*$*$ end of routine -
 insert your code only before this line         *-*
  ENDMETHOD.                    "compute_0PASSFORM
```

The corresponding inversion routine is as follows:

```
*$*$ begin of inverse routine -
 insert your code only below this line*-*
    DATA l_r_set        TYPE REF TO cl_rsmds_set.
    IF i_r_selset_outbound->is_universal( ) EQ rsmds_c_boolean-true.

*     If query requests all values for characteristic 0PASSNAME
*     request also all values from source field PASSNAME
      c_r_selset_inbound = cl_rsmds_set=>get_universal_set( ).
      c_exact = rs_c_true. "Inversion is exact

    ELSE.
      TRY.
          IF me->p_r_set_mrs IS INITIAL.
*           Create set for condition PASSFORM = 'FRAU'
            me->p_r_set_mrs = i_r_universe_inbound-
>create_set_from_string( 'PASSFORM = ''FRAU''' ).
          ENDIF.
          IF me->p_r_set_mr IS INITIAL.
*           Create set for condition PASSFORM = 'HERR'
            me->p_r_set_mr = i_r_universe_inbound-
>create_set_from_string( 'PASSFORM = ''HERR''' ).
          ENDIF.
          IF me->p_r_set_space IS INITIAL.
*           Create set for condition NOT ( PASSFORM = 'FRAU' OR PASSF
ORM = 'HERR' )
            l_r_set           = me->p_r_set_mr->unite( me-
>p_r_set_mrs ).
            me->p_r_set_space = l_r_set->complement( ).
          ENDIF.

*         Compose inbound selection
          c_r_selset_inbound = cl_rsmds_set=>get_empty_set( ).
*         Check if outbound selection contains value 'MR'
          IF i_r_selset_outbound-
>contains( 'MR'  ) EQ rsmds_c_boolean-true.
            c_r_selset_inbound = c_r_selset_inbound->unite( me-
>p_r_set_mr ).
          ENDIF.
*         Check if outbound selection contains value 'MRS'
          IF i_r_selset_outbound-
>contains( 'MRS' ) EQ rsmds_c_boolean-true.
            c_r_selset_inbound = c_r_selset_inbound->unite( me-
>p_r_set_mrs ).
          ENDIF.
*         Check if outbound selection contains initial value
          IF i_r_selset_outbound-
>contains( space ) EQ rsmds_c_boolean-true.
            c_r_selset_inbound = c_r_selset_inbound->unite( me-
>p_r_set_space ).
          ENDIF.
          c_exact = rs_c_true. "Inversion is exact

        CATCH cx_rsmds_dimension_unknown
              cx_rsmds_input_invalid
              cx_rsmds_sets_not_compatible
              cx_rsmds_syntax_error.

*         Normally, should not occur
```

```
*         If the exception occurs request all values from source
*         for this routine to be on the save side
          c_r_selset_inbound = cl_rsmds_set=>get_universal_set( ).
          c_exact = rs_c_false. "Inversion is no longer exact

      ENDTRY.
    ENDIF.

*   Finally, add (optionally) further code to transform outbound proj
ection
*   to inbound projection
*   Check if outbound characteristic 0PASSFORM (field name PASSFORM)
*   is requested for the drilldown state of the query
    READ TABLE i_th_fields_outbound
         WITH TABLE KEY segid     = 1           "Primary segment
                        fieldname = 'PASSFORM'
         TRANSPORTING NO FIELDS.
    IF sy-subrc EQ 0.
*     Characteristic 0PASSFORM is needed
*     ==> request (only) field PASSFORM from the source for this rout
ine
      DELETE c_th_fields_inbound
        WHERE NOT ( segid     EQ 1          OR
                    fieldname EQ 'PASSFORM'    ).
    ELSE.
*     Characteristic 0PASSFORM is not needed
*     ==> don't request any field from source for this routine
      CLEAR c_th_fields_inbound.
    ENDIF.

*$*$ end of inverse routine -
 insert your code only before this line *-*
  ENDMETHOD.                    "invert_0PASSFORM
ENDCLASS.                   "routine IMPLEMENTATION
```

4.1.7.5.6 Regular Expressions in Routines

Use

You can use regular expressions in routines.

A regular expression [External] (abbreviation: RegExp or Regex) is a pattern of literal and special characters which describes a set of character strings. In ABAP, you can use regular expressions in the FIND and REPLACE statements, and in classes CL_ABAP_REGEX and CL_ABAP_MATCHER. For more information, see the ABAP key word documentation in the ABAP Editor. This documentation describes the syntax of regular expressions and you can test regular expressions in the ABAP Editor.

Example

This section provides sample code to illustrate how you can use regular expressions in routines.

```
REPORT z_regex.
```

```
DATA: l_input TYPE string,
      l_regex TYPE string,
      l_new   TYPE string.

* Example 1: Insert thousand separator

l_input = '12345678'.
l_regex = '([0-9])(?=([0-9]{3})+(?![0-9]))'.
l_new = '$1,'.

WRITE: / 'Before:', l_input.    "12345678

REPLACE
  ALL OCCURRENCES OF
  REGEX l_regex
  IN l_input WITH l_new.

WRITE: / 'After:', l_input.     "12,345,678

* Example 2: Convert date in US format to German format

l_input = '6/30/2005'.
l_regex = '([01]?[0-9])/([0-3]?[0-9])/'.
l_new = '$2.$1.'.

WRITE: / 'Before:', l_input.    "6/30/2005

REPLACE
  ALL OCCURRENCES OF
  REGEX l_regex
  IN l_input WITH l_new.

WRITE: / 'After:', l_input.     "30.6.2005

* Example 3: Convert external date in US format to internal date

DATA: matcher   TYPE REF TO cl_abap_matcher,
      submatch1 TYPE string,
      submatch2 TYPE string,
      match     TYPE c.

l_input = '6/30/2005'.
l_regex = '([01]?)([0-9])/([0-3]?)([0-9])/([0-9]{4})'.

matcher = cl_abap_matcher=>create( pattern     = l_regex
                                  text        = l_input ).

match = matcher->match( ).

TRY.
    CALL METHOD matcher->get_submatch
      EXPORTING
        index    = 1
      RECEIVING
        submatch = submatch1.
  CATCH cx_sy_matcher.
ENDTRY.

TRY.
    CALL METHOD matcher->get_submatch
      EXPORTING
        index    = 3
      RECEIVING
        submatch = submatch2.
```

```
  CATCH cx_sy_matcher.
ENDTRY.

IF submatch1 IS INITIAL.
  IF submatch2 IS INITIAL.
    l_new = '$5\0$2\0$4'.
  ELSE.
    l_new = '$5\0$2$3$4'.
  ENDIF.
ELSE.
  IF submatch2 IS INITIAL.
    l_new = '$5$1$2\0$4'.
  ELSE.
    l_new = '$5$1$2$3$4'.
  ENDIF.
ENDIF.

WRITE: / 'Before:', l_input.    "6/30/2005

REPLACE
  ALL OCCURRENCES OF
  REGEX l_regex
  IN l_input WITH l_new.

WRITE: / 'After:', l_input.     "20050630
```

4.1.7.6 InfoSource

Definition

A non-persistent structure consisting of InfoObjects for joining two transformations.

Use

You use InfoSources if you want to run two (or more) sequential transformations in the data flow, without storing the data again.

If you do not have transformations that run sequentially, you can model the data flow without InfoSources. In this case, the data is written straight to the target from the source using a transformation.

However, it may be necessary to use one or more InfoSources for semantic or complexity reasons. For example, you need one transformation to ensure the format and the assignment to InfoObjects and an additional transformation to run the actual business rules. If this involves complex inter-dependent rules, it may be useful to have more than one InfoSource.

See also Recommendations for Using InfoSources [Page 262].

Structure

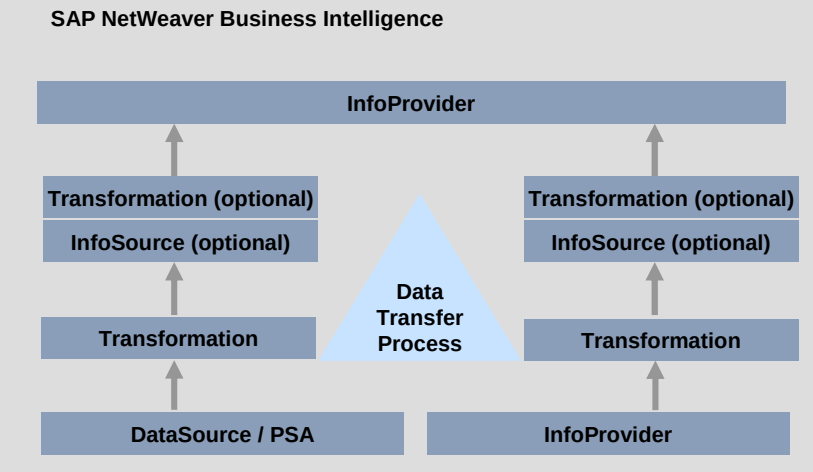
In contrast to 3.x InfoSources, as of Release SAP NetWeaver BI 2004s, an InfoSource behaves like an InfoSource with flexible update. See 3.x InfoSource [Page 270].

The data in an InfoSource is updated to an InfoProvider using a transformation.

You can define the InfoObjects of the InfoSource as keys. These keys are used to aggregate the data records during the transformation.

Integration

The following figure shows how InfoSources are integrated into the data flow:



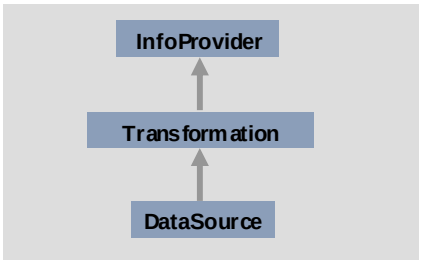
You create the data transfer process from a DataSource to an InfoProvider. Since InfoSources are not persistent data stores, they cannot be used as targets of the data transfer process. You create transformations for an InfoProvider (as the target) with an InfoSource (as the source), and for an InfoSource (as the target) with a DataSource (as the source).

4.1.7.6.1 Recommendations for Using InfoSources

This section outlines three scenarios for using InfoSources. The decision to use an InfoSource depends on how the effort involved in maintaining the InfoSource and any potential changes in the scenario can be minimized.

1. Data Flow Without an InfoSource:

The DataSource is connected directly to the target by means of a transformation.



Since there is only one transformation, performance is better.

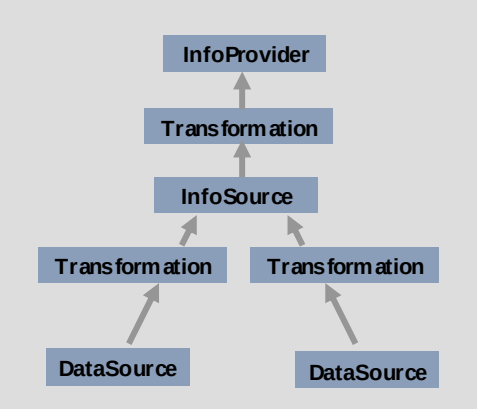
However, if you want to connect multiple DataSources with the same structure to the target, this can result in additional maintenance effort for the transformation, since you need to create a similar transformation for each DataSource.

You can avoid this if the DataSource is the same, it just appears in different source systems. In this case, you can use source system mapping when you transport to the target system so

that only one transformation has to be maintained in the test system. The same transformation is created automatically for each source system in the production system.

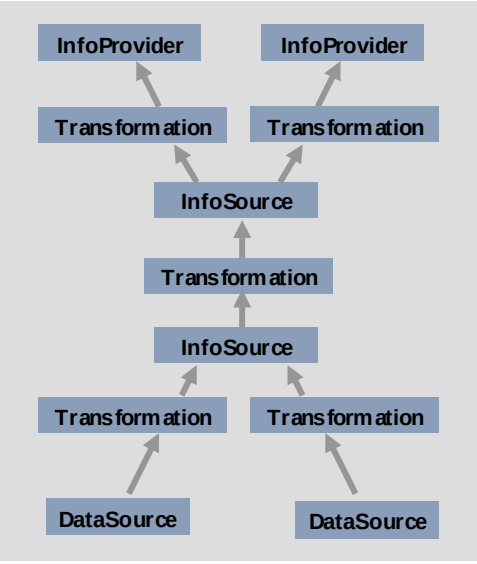
2. Data Flow with One InfoSource

The DataSource is connected to the target by means of an InfoSource. There is one transformation between the DataSource and the InfoSource and one transformation between the InfoSource and the target.



We recommend that you use an InfoSource if you want to connect a number of different DataSources to a target and the different DataSources have the same business rules. In the transformation, you can align the format of the data in DataSource with the format of the data in the InfoSource. The required business rules are applied in the subsequent transformation between the InfoSource and the target. You can make any changes to these rules centrally in this one transformation, as required.

3. Data Flow with Two InfoSources



We recommend that you use this type of data flow if your data flow not only contains two different sources, but the data is to be written to multiple identical (or almost identical) targets. The required business rules are executed in the central transformation so that you only have

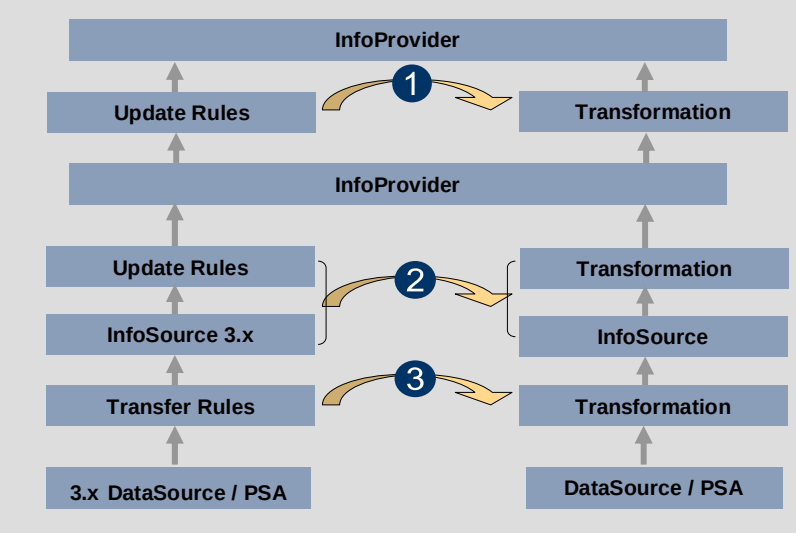
to modify the one transformation in order to change the business rules. You can connect sources and targets that are independent of this transformation.

4.1.7.7 Migration of Update Rules, 3.x InfoSources, and Transfer Rules

Use

You can create a transformation using update rules or transfer rules. In doing so, the corresponding 3.x InfoSource is converted into a (new) InfoSource. This allows you to migrate existing objects to the new transformation concept after you upgrade.

When you create the transformation and the (new) InfoSource, the system retains the update rules, 3.x InfoSources, and transfer rules. To ensure that the loading process is performed using the transformation and not the update rules or transfer rules, data has to be loaded using a data transfer process.



Procedure

1. Data Flow Between Two InfoProviders: Creating Transformations Using Update Rules

1. You are in the *Modeling* functional area of the Data Warehousing Workbench. In the context menu of the update rule you want to convert, choose *Additional Functions* → *Create Transformation*. No export DataSource is now needed for the data flow between two InfoProviders (Myself Data Mart). The transformation for which you create another data transfer process is sufficient.

2. Data Flow Between DataSource and InfoProvider: Creating Transformations Using Update Rules

1. You are in the *Modeling* functional area of the Data Warehousing Workbench. In the context menu of the update rule you want to convert, choose *Additional Functions* → *Create Transformation*.
2. You can choose whether you want to create a new InfoSource or use an existing one.

3. Choose *Okay*. The system generates a log for the conversion. The InfoSource is activated immediately; the transformation is saved without being activated. If you want to use routines in the rules, you might have to edit the transformation manually.

3. Creating Transformations from Transfer Rules

1. You are in the *Modeling* functional area of the Data Warehousing Workbench. In the context menu of the transfer rule you want to convert, choose *Additional Functions* → *Create Transformation*.
2. You can choose whether you want to create a new InfoSource or use an existing one. If you have already converted the update rules, a converted version of the related InfoSource already exists.
3. Choose *Okay*. The system generates a log for the conversion. The transformation is saved without being activated. If you want to use routines in the rules, you might have to edit the transformation manually.

4. Editing Transformations

1. Routines are copied straight to the global part 2. Using a PERFORM, the routine for the converted rule is called from the routine for the transformation. Comments on the conversion are retained in the routines. You can use these comments to modify the routine to improve performance.

   If you programmed the routine dynamically, you should check it by performing a before-after check. Since the fields of the source structure are removed during migration, errors that cannot be checked by the system could occur in the converted routine. If fields that are used in the routine are not filled, an error occurs.

   More information: Differences in Routine Concepts [Page 265]
2. Return tables in routines cannot be converted. We recommend that you use an end routine instead.
3. Inversion routines in transfer rules are not converted. If the transfer rules contain inversion routines, you have to recreate these in the transformation.

   The following example may be of use: Example for Migration of an Inversion Routine [Page 266]
4. Activate your transformation.

Result

You can create a data transfer process for the new objects.

4.1.7.7.1 Differences in Routine Concepts

The way in which routines can be implemented changes when the programming language for routines is converted from ABAP to ABAP objects.

The following table provides an overview of the special features regarding ABAP form routines for the update and transfer rules in comparison to the routines in the transformation.

Form Routine for Update/Transfer Rule	Routine for Transformation
Parameter COMM_STRUCTURE	SOURCE_FIELDS

Parameter ABORT <> 0	RAISE EXCEPTION TYPE CX_RSROUT_ABORT.
Parameter RETURNCODE <> 0	RAISE EXCEPTION TYPE CX_RSROUT_SKIP_RECORD (for key fields) or RAISE EXCEPTION TYPE CX_RSROUT_SKIP_VALUE (for non-key fields)
Subprograms are included in the global part of the routine using an INCLUDE	You cannot use INCLUDES. You can convert these subprograms in the following ways: 1. Convert the subprograms into global, static methods. 2. Create a subroutine pool in the ABAP editor and execute these subprograms using PERFORM SUBROUTINE. 3. Define a function module that has the logic of the subprogram. Function modules, methods or external subprograms can be called in the local part of the routine.
STATICS statement	The STATICS statement is not permitted in instance methods. Declared static attributes of the class can be used with CLASS DATA instead.
Addition OCCURS when the internal table is created	The OCCURS addition is not permitted. You use the DATA statement to declare a standard table instead.
Internal table with header row	You cannot use an internal table with a header row. You create an explicit work area with the LINE OF addition of statements TYPES, DATA and so on to replace the header row.
Direct operations such as INSERT itab, APPEND itab on internal tables	You have to use a work area for statements of this type.

4.1.7.7.2 Example for Migration of an Inversion Routine

This example is a nearly universally applicable model that shows how to migrate an inversion routine from a transfer rule.

In the routine, the German keys 'HERR' and 'FRAU' in the target characteristic are mapped to the English keys 'MR' and 'MRS' of the field PASSFORM (form of address) of the source. All other values from the source field are mapped to the initial value.

A further example does the same, but is optimized for the new method interface. Compare this with Example for Inversion Routine [Page 256].

```
*$*$ begin of inverse routine -
 insert your code only below this line*-*
*   Simulate 3.X interface by defining variables of the same name
*   and the same type as the FORM routine parameters of the 3.X routi
ne

    DATA:
      i_rt_chavl_cs      TYPE rsarc_rt_chavl,
      i_thx_selection_cs TYPE rsarc_thx_selcs,
      c_t_selection      TYPE sbiwa_t_select,
      e_exact            TYPE rs_bool.

    DATA:
      l_tr_dimensions    TYPE rsmds_tr_dimensions,              "tabl
e of dimension references
      l_r_dimension      LIKE LINE OF l_tr_dimensions,          "dime
nsion reference
      l_dimname          TYPE rsmds_dimname,                    "dime
nsion name
      l_sx_selection_cs  LIKE LINE OF i_thx_selection_cs,       "work
 area for single characteristc RANGE table
      l_r_universe       TYPE REF TO cl_rs_infoobject_universe. "refe
rence for InfoObject universe

    TRY.

*       Transform selection set for outbound (=target)
*       characteristic 0PASSFORM to RANGE table
        CALL METHOD i_r_selset_outbound->to_ranges
          CHANGING
            c_t_ranges = i_rt_chavl_cs.

*       Transform complete outbound selection set to extended RANGES
table
*       (The following step can be skipped if I_THX_SELECTION_CS is n
ot used
*        by the 3.X implemention as it is the case here)

*       Get reference to InfoObject universe (singleton)
        l_r_universe = cl_rs_infoobject_universe=>get_instance( ).
*       Get all dimensions (i.e. fields) from outbound selection whic
h are
*       restricted
        l_tr_dimensions = i_r_selset_outbound_complete-
>get_dimensions( ).
        LOOP AT l_tr_dimensions INTO l_r_dimension.
          CLEAR l_sx_selection_cs.
*         Get dimension name (= field name)
          l_dimname = l_r_dimension->get_name( ).
*         Transform dimension name to InfoObject name
          l_sx_selection_cs-chanm = l_r_universe->dimname_to_iobjnm(
          l_dimname ).
*         Project complete outbound selection set to current dimensio
```

```
n and
*         and convert to RANGE table representation
          CALL METHOD i_r_selset_outbound_complete->to_ranges
            EXPORTING
              i_r_dimension = l_r_dimension
            CHANGING
              c_t_ranges    = l_sx_selection_cs-rt_chavl.
          APPEND l_sx_selection_cs TO i_thx_selection_cs.
        ENDLOOP.

*$*$ Insert your 3.X implementation between here ... *---------------
--*

        DATA:
          l_s_selection      LIKE LINE OF c_t_selection.

        l_s_selection-fieldnm = 'PASSFORM'.
        CLEAR l_s_selection-high.
        IF space IN i_rt_chavl_cs.
*         Select all values from source except ...
          l_s_selection-sign   = 'E'.
          l_s_selection-option = 'EQ'.
          IF NOT 'MR'  IN i_rt_chavl_cs.
*           ... 'HERR' and ...
            l_s_selection-low    = 'HERR'.
            APPEND l_s_selection TO c_t_selection.
          ENDIF.
          IF NOT 'MRS' IN i_rt_chavl_cs.
*           ... 'FRAU'
            l_s_selection-low    = 'FRAU'.
            APPEND l_s_selection TO c_t_selection.
          ENDIF.
        ELSE.
          IF 'MR'  IN i_rt_chavl_cs.
            l_s_selection-sign   = 'I'.
            l_s_selection-option = 'EQ'.
            l_s_selection-low    = 'HERR'.
            APPEND l_s_selection TO c_t_selection.
          ENDIF.
          IF 'MRS' IN i_rt_chavl_cs.
            l_s_selection-sign   = 'I'.
            l_s_selection-option = 'EQ'.
            l_s_selection-low    = 'FRAU'.
            APPEND l_s_selection TO c_t_selection.
          ENDIF.
          IF c_t_selection IS INITIAL.
*           Other values cannot occur as transformation result
*           ==> source will not contribute to query result
*                 with any record
*           ==> return empty selection (e.g. include and exclude init
ial value)
            l_s_selection-sign   = 'I'.
            l_s_selection-option = 'EQ'.
            CLEAR: l_s_selection-low, l_s_selection-high.
            APPEND l_s_selection TO c_t_selection.
            l_s_selection-sign   = 'E'.
            APPEND l_s_selection TO c_t_selection.
          ENDIF.
```

```
        ENDIF.
        e_exact = rs_c_false.  "This inversion is exact

*$*$ ... and here *-------------------------------------------------
--*
*       Convert 3.X inversion result to new method interface
        c_r_selset_inbound = i_r_universe_inbound-
>create_set_from_ranges(
                                i_fieldname_dimension = 'FIELDNM'
                                i_t_ranges            = c_t_selection
).
        c_exact = e_exact.

      CATCH cx_rsmds_input_invalid
            cx_rsmds_input_invalid_type.

*       Should not occur
*       If the exception occurs request all values from source
*       for this routine to be on the save side
        c_r_selset_inbound = cl_rsmds_set=>get_universal_set( ).
        c_exact = rs_c_false. "Inversion is no longer exact

    ENDTRY.

*   Finally, add (optionally) further code to transform outbound proj
ection
*   to inbound projection
*
*   Please note:
*
*   In 3.X you did this mapping before entering the source code edito
r.
*   For the transformation in SAP Netweaver BI 7.0 the passed inbound
 projection
*   C_TH_FIELDS_INBOUND already contains all fields from the source s
tructure
*   which are required by this routine according to the rule definiti
on.
*   Remove lines from this internal table if the corresponding field
*   is not requested for the query.

*   Check if outbound characteristic 0PASSFORM (field name PASSFORM)
*   is requested for the drilldown state of the query
    READ TABLE i_th_fields_outbound
         WITH TABLE KEY segid     = 1           "Primary segment
                        fieldname = 'PASSFORM'
         TRANSPORTING NO FIELDS.
    IF sy-subrc EQ 0.
*     Characteristic 0PASSFORM is needed
*     ==> request (only) field PASSFORM from the source for this rout
ine
      DELETE c_th_fields_inbound
        WHERE NOT ( segid     EQ 1          OR
                    fieldname EQ 'PASSFORM'    ).
    ELSE.
*     Characteristic 0PASSFORM is not needed
*     ==> don't request any field from source for this routine
      CLEAR c_th_fields_inbound.
```

```
    ENDIF.

*$*$ end of inverse routine -
 insert your code only before this line *-*
  ENDMETHOD.                    "invert_0PASSFORM
ENDCLASS.                     "routine IMPLEMENTATION
```

4.1.7.8 Old Transformation Concept

The transformation process allows you to define rules for consolidating, cleansing, and integrating data. You can define semantic keys for the aggregation.

In releases before SAP NetWeaver 7.0, the central object for the transformation is the InfoSource. The individual fields of the DataSource are assigned to the relevant InfoObjects in the InfoSource. The data can then be transformed using transfer rules. The update rules then specify how the data (key figures, time characteristics, characteristics) is updated into the InfoProvider from the communication structure of an InfoSource. The data can also be transformed in the update rules.

You can continue to use this concept, however we recommend that you change to the new transformation concept as of SAP NetWeaver 7.0. The new concept offers enhanced functionality, better performance, improved log functions and better usability. In addition, the new concept will continue to be developed, whereas the old functionality will not be developed further.

In the new transformation concept, you no longer require two different rules (a transfer rule and an update rule). You only need the transformation rules. You edit the transformation rules on a more intuitive graphical user interface. InfoSources are no longer mandatory; they are optional and are only required for certain functions. Transformations also provide additional functions such as quantity conversion and the option of creating an end routine or expert routine.

4.1.7.8.1 3.x InfoSource

Definition

3.x InfoSources specify the set of all data available for a business transaction or a type of business transaction (for example, cost center accounting).

3.x InfoSources are sets of logically-related information, summarized into a single unit. They serve to stage consolidated data that can be updated into additional InfoProviders. 3. x InfoSources can contain either transaction data or master data (attributes, texts, and hierarchies).

They are always sets of logically-related InfoObjects [Page 57] that are available in the form of a communication structure.

A new type of InfoSource is available as of SAP NetWeaver 7.0. You can continue to create and use 3.x InfoSources, however we recommend that you use the new InfoSource concept with the new transformation concept.

In the Data Warehousing Workbench, the ▫ icon before the description identifies an object that is available for the new concept.

Use

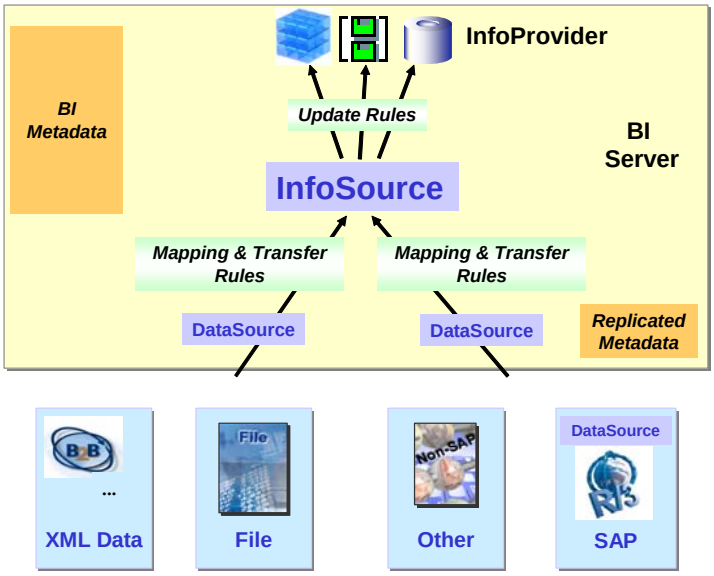
In the BI system, a DataSource is assigned to an InfoSource. If fields that logically belong together exist in different source systems, they can be grouped together in a single InfoSource in the BI system by assigning multiple DataSources to one InfoSource.

In transfer rule maintenance [Page 281], individual DataSource fields are assigned to the corresponding InfoObject of the InfoSource. Here you can also specify how the data of a DataSource is transferred to the InfoSource. The uploaded data is transformed using transfer rules. An extensive library of transformation functions that contain business logic can be used here to clean up data and allow it to be analyzed. The rules can be applied simply, without coding, by using formulas.

The transfer structure is used to transfer data into the BI system. The data is transferred 1:1 from the transfer structure [Page 555] of the source system into the transfer structure of the BI system.

Integration

If logically-related fields exist in different source systems, they can be grouped together into a single InfoSource in the BI system. The source system release is not important here.



If you have an InfoSource with flexible update, you use update rules [Page 300] to update data from the communication structure of the InfoSource into further InfoProviders. InfoSources with direct update allow master data to be written to the master data tables directly (without update rules).

InfoSources are listed under an application component in the InfoSource tree of the Data Warehousing Workbench.

4.1.7.8.1.1 3.x InfoSource Types

There are two types of 3.x InfoSources:

- InfoSources with flexible updating
- InfoSources with direct updating

In both cases, uploaded data is transformed using the transfer rules, which have been created for the current combination of InfoSource and source system and for each InfoObject of the communication structure. An InfoProvider can be supplied by multiple InfoSources, which in turn can be supplied by multiple source systems.

An InfoSource for hierarchies can only be supplied by one source system.

For characteristics, attributes, or texts, a combination of flexible and direct updating is only possible for different source systems.

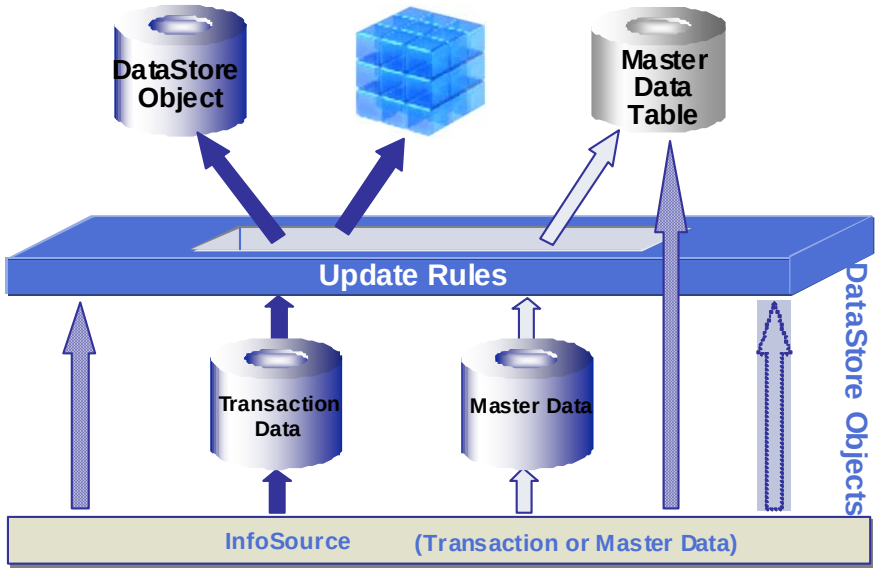
InfoSources with Flexible Updating

For an InfoSource with flexible updating, the data from the communications structure is loaded into the data targets (InfoCubes, DataStore objects, master data) using update rules [Page 300]. Several data targets can be supplied by one InfoSource. The InfoSource can contain transaction data as well as master data.

This function is not available for hierarchies.

Before Release 3.0A, only transaction data could be updated flexibly and it was only possible to update master data directly. Master data InfoSources were therefore distinguished from transaction data InfoSources. This is no longer the case as of Release BW 3.0A, since both transaction data and master data can be updated flexibly. You therefore cannot immediately see if an InfoSource with flexible updating handles transaction data or master data. You should therefore specify this in the description of an InfoSource.

Flexible Updating – Data Flow



You have the following update options:

- The data from InfoSources with master data or transaction data can be stored directly in DataStore objects.
- You can then use update rules to update from DataStore objects into further DataStore objects, InfoCubes, or master data tables.
- It is also possible to update into InfoCubes or master data tables without having to switch between DataStore objects.

InfoSources with Direct Updating

Using an InfoSource with direct updating, master data (characteristics with attributes, texts, or hierarchies) of an InfoObject can be updated directly (without update rules, only using transfer rules) into the master data table. To do this you must assign it an application component. The system displays the characteristic in the InfoSource tree in the Data Warehousing Workbench. You can assign DataSources and source systems to the characteristic from there. You can then also load master data, texts, and hierarchies for the characteristic.

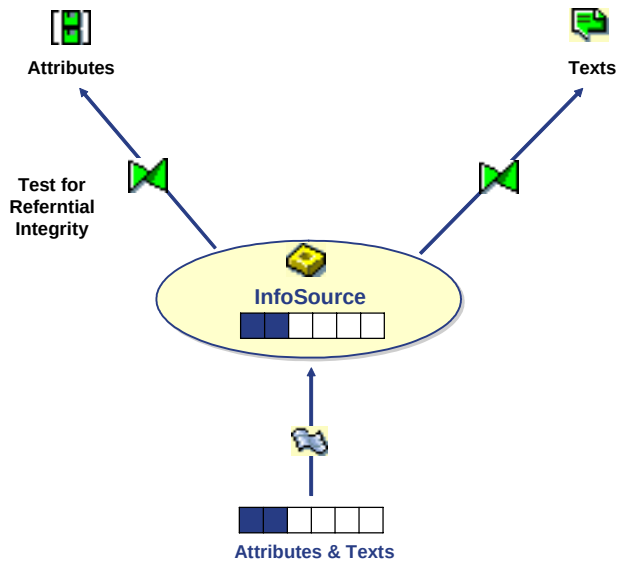
You cannot use an InfoObject as an InfoSource with direct updating if:

- The characteristic you want to modify is characteristic 0SOURSYSTEM (source system ID).
- The characteristic has neither master data nor texts nor hierarchies. It is therefore impossible to load data for the characteristic.
- The characteristic that you want to modify turns out not to be a characteristic, but a unit or a key figure.

To generate an export DataSource for a characteristic, the characteristic must also be an InfoSource with direct updating.

4.1.7.8.1.2 Scenarios for Flexible Updating

1. Attributes and texts are delivered together in a file:

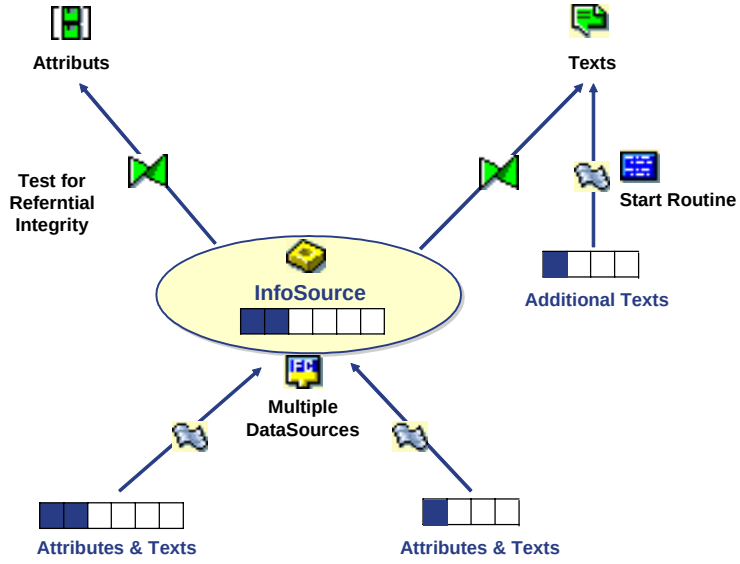


Your master data, attributes, and texts are available together in a flat file. They are updated by an InfoSource with flexible updating in additional InfoObjects. In doing so, texts and attributes can be separated from each other in the communication structure.

Flexible updating is **not** necessary if:

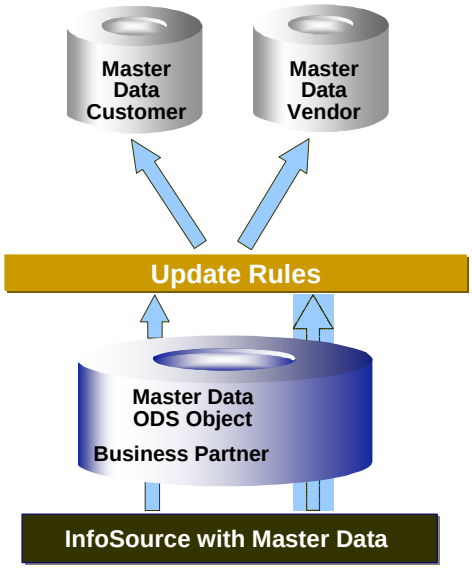
- texts and attributes are available in separate files/DataSources. In this case, you can choose direct updating if additional transformations using update rules are not necessary.

2. Attributes and texts come from several DataSources:



This scenario is similar to the one described above, only slightly more complex. Your master data comes from two different source systems and delivers attributes and texts in flat files. They are grouped together in an InfoSource with flexible updating. Attributes and texts can be separated in the communication structure and are updated further in InfoObjects. The texts or attributes from both source systems are located in these InfoObjects.

3. Master data in the ODS layer:



A master data InfoSource is updated to a master data ODS object business partner with flexible updating. The data can now be cleaned and consolidated in the ODS object before being re-read. This is important when the master data frequently changes.

These cleaned objects can now be updated to further ODS Objects. The data can also be selectively updated using routines in the update rules. This enables you to get views of selected areas. The data for the business partner is divided into customer and vendor here.

Instead you can update the data from the ODS object in InfoObjects as well (with attributes or texts). When doing this, be aware that loading of deltas takes place serially. You can ensure this when you activate the automatic updates in ODS object maintenance or when you perform the loading process using a process chain (see also Including ODS Objects in a Process Chain [External]).

A master data ODS object generally makes the following options available:

- It displays an additional level on which master data from the whole enterprise can be consolidated.
- ODS objects can be used as a validation table for checking the referential integrity [Page 279] of characteristic valuables in the update rules.
- It can serve as a central repository for master data, in which master data is consolidated from various systems. They can then be forwarded to further BW systems using the Data Mart.

4.1.7.8.1.3 Creating InfoSources (SAP Source System)

Use

Instead of creating a new InfoSource, you can copy one from SAP Business Content [Page 435].

Procedure

Choose the InfoSource tree of the Data Warehousing Workbench to create InfoSources for an SAP source system. From the context menu of the affected application component, choose *Additional Functions → Create InfoSource 3.x*.

1. Select the type [Page 272].
2. Under *InfoSource,* enter the technical name of the InfoSource, and then a description.

   You can also use an existing InfoSource as a template.
3. Assign a source system to the InfoSource and confirm.
4. From the proposal list, select the DataSource from which transaction data is to be loaded.

   Transfer structure maintenance automatically appears.

   The system automatically offers you suitable transfer rules, but you can modify these.
5. Maintain the transfer structure [Page 280]. Assign InfoObjects to the fields of the DataSource.
6. The communications structure [Page 277] is adjusted automatically, but you can also include more fields. Activate your selection.
7. Maintain the transfer rules [Page 281].
8. Activate the InfoSource.

Result

The InfoSource is now saved and active.

See also:

Maintaining InfoSources (Flat Files) [Page 286]

Maintaining InfoSources (External System) [Page 300]

4.1.7.8.1.3.1 Communication Structure

Definition

The communication structure is localized in the SAP Business Information Warehouse and displays the structure of the InfoSource. It contains all of the InfoObjects belonging to the InfoSource of the SAP Business Information Warehouse.

Use

Data is updated in the data targets of this structure. In this way, the system always accesses the actively saved version of the communication structure.

In the transfer rules maintenance, you determine whether the communication structure is filled with fixed values from the transfer structure fields, by means of a formula or using a local conversion routine.

Conversion routines are ABAP programs that you can create yourself. The routine always refers to just one InfoObject of the transfer structure.

4.1.7.8.1.3.1.1 Maintaining Communications Structures with Flexible Updating

Use

Assign the fields of a DataSource to the corresponding InfoObjects of the InfoSource. Although the DataSource contains source system fields that belong together, you can combine fields that logically belong together from several DataSources, from various source systems in the communication structure.

Prerequisites

You have created an InfoSource with flexible updating.

Procedure

You can get to the maintenance of the communication structure using the InfoSource tree of the Administrator Workbench.

1. Choose *Your Application Components* → *Your InfoSource* → *Context menu (right mouse click)* → *Change*.

   - You can enter the required InfoObjects directly into the left-hand column of the communication structure. You can also select InfoObjects using F4 Help, or create new characteristics and key figures by using the toolbar.
   - If you have already assigned a source system and a communication structure already exists with transfer rules and transfer structure, then you are displayed the InfoObjects from the transfer structure in the template. You can select InfoObjects and transfer or remove them from the template into the communication structure using the arrow.

2. You can also define that the referential integrity should be checked. You can set the InfoObjects that should be checked. A check against the master data ODS object, if it exists, always makes sense. Alternatively you can check against the master data table.

   Also refer to Checking for Referential Integrity [Page 279].

   In the InfoObject maintenance you can define the ODS object against which you want to check. See also Tabstrip: Master Data/Texts [Page 80].

3. Check your entries and save.

   You can only use the InfoObjects of the communication structure in the update rules if you have activated your entries.

   If a communication structure already existed in an active version, then the system always reverts back to this when maintaining the update rules. The

version of the communication structure, which was created by a simple save, is not used.

InfoObjects of the communication structure, which are used in the update rules or in the transfer rules, cannot be removed from the communication structure.

If an InfoObject is used, the corresponding fields in the communication structure maintenance are highlighted.

Result

You have determined InfoObjects that can be updated in the data targets.

4.1.7.8.1.3.1.2 Maintaining Communication Structures with Direct Updating

Use

Assign the fields of a DataSource to the corresponding InfoObjects of the InfoSource. Although the DataSource contains source system fields that belong together, you can combine fields that logically belong together from several DataSources, from various source systems in the communication structure.

Prerequisites

You have created an InfoSource with direct updating. You have assigned a source system and a data source to it.

Procedure

After you have assigned the source system, you automatically reach the transfer structure maintenance.

1. Maintain the transfer structure [Page 280] and the transfer rules [Page 281] in the lower half of the screen.

   In the upper half of the screen, you can view the communication structure. The system automatically generates this.

   Depending on whether you have specified a DataSource for attributes or texts, the communication structure contains the attributes or text fields next to the corresponding InfoObject.

   The upper half of the screen (the communication structure) is hidden if you have specified a DataSource for hierarchies to be transferred via IDoc. The communication structure is generated for hierarchies that are transferred using a PSA.

2. Activate your settings.

4.1.7.8.1.3.1.3 Checking for Referential Integrity

Use

The check for referential integrity occurs for transaction data and master data if they are flexibly updated. You determine the valid InfoObject values.

Prerequisites

The check for referential integrity functions only in conjunction with the function *Error Handling* on the scheduler tab page *Update*.

See also Handling Data Records with Errors [Page 373].

In order to use the check for referential integrity, you have to choose the option *Always Update Data...* . If you choose the option *Do Not Update Data...*, you override the check for referential integrity. This is valid for master data (with flexible updating) as well as for transaction data.

Difference in Treating Data Records with Errors

Checking for Referential Integrity	Treating data records with errors
For all InfoProviders	For all InfoProviders
Check in the transfer rules	Check according to update rules for each InfoProvider
Only for selected InfoObjects	For all InfoObjects
Error Handling	Terminates after first incorrect record
Possible for all DataStore objects	BW 2.0: Only for DataStore objects for which BEx Reporting is switched on
Check against master data table or against a DataStore object possible	Checked against master data table

Features

The verification occurs after filling the communication structure and before filling the update rules. What is displayed in the InfoObject metadata is checked against the master data table (meaning the SID table) or against another DataStore object.

If you create a DataStore object for checking the characteristic values in a characteristic, in the update rules, and in the transfer rules, the valid values for the characteristic are determined from the DataStore object and not from the master data.

4.1.7.8.1.3.2 Transfer Structure in Data Flow 3.x

Definition

The transfer structure is the structure in which the data is transported from the source system into BI.

It is a selection of DataSource fields from a source system.

Use

The transfer structure provides BI with all the source system information available for a business process.

An InfoSource 3.x in BI needs at least a DataSource 3.x for data extraction. In an SAP source system, DataSource data that logically belongs together is staged in a flat structure, the extraction structure. In the source system, you are able to filter and enhance the extraction structure in order to determine the DataSource fields.

In the transfer structure maintenance in BI, you determine which fields of the DataSource 3.x are to be transferred to BI. When you activate the transfer rules in BI, a transfer structure identical to the one in BI is created in the source system from the DataSource fields.

This data is transferred 1:1 from the transfer structure of the source system into the BI transfer structure, and is then transferred into the BI communication structure using the transfer rules.

A transfer structure always refers to a DataSource from a source system and to an InfoSource in BI.

If you choose *Create Transfer Rules* from the DataSource or the InfoSource in an object tree of the Data Warehousing Workbench, the transfer structure maintenance appears.

4.1.7.8.1.3.2.1 Maintaining Transfer Structures

Use

In the transfer structure maintenance, you specify which fields for a DataSource are to be transferred into a communication structure.

Prerequisites

You have created an InfoSource. You have maintained a communications structure.

A maintained communication structure is required for the procedure described in the following. It is however also possible for you to create the InfoSource first, then maintain the communication structure and finally assign the source system.

Procedure

1. Select a source system.
2. Select a DataSource that has already been connected using F4 Help or add a new DataSource using *Assign DataSource.*

   The fields for the DataSource are displayed in the right half of the tabstrip transfer structure. These are transferred to the left half in the transfer structure by default.
3. Maintain the transfer rules [Page 281]

4.1.7.8.1.3.3 Processing Transfer Rules

Use

When you have maintained the transfer structure and the communication structure, you use the transfer rules to determine how you want the transfer structure fields to be assigned to the communication structure InfoObjects. You can arrange for a 1:1 assignment. You can also fill InfoObjects using routines, formulas, or constants.

You need not assign InfoObjects to each field of the transfer structure. If you only need a field for entering a routine or for reading from the PSA, you need not create an InfoObject.

However, you must keep the following in mind: When you load data from non-SAP systems, the information from the InfoObject is used as the basis for converting the key figures into the SAP format. In this case you must assign an InfoObject to the field. Otherwise wrong numbers might be loaded or the numbers might be displayed incorrectly in the reports. For more information, also see Conversion Routines in BW [Page 98].

Prerequisites

Before you are able to maintain the transfer rules for an InfoSource, you must assign a source system to the InfoSource and create a communication structure.

Procedure

1. You maintain the transfer rules in the InfoSource tree of the Administrator Workbench.

   For InfoSources, choose *Your Application Components* → *Your InfoSource* → *Context menu (right mouse-click)* → *Change*.

2. Select a transfer method. We recommend the PSA transfer method. In the Scheduler you also have other options for data updating. See also Tab Page: Processing [Page 350].

3. The transfer structure is displayed in the right half of the screen along with the selected DataSource fields.

   The system uses the data elements to help it suggest InfoObjects that could be assigned to the corresponding fields of the DataSource. These suggested InfoObjects are displayed in the left column of the transfer structure

   The fields for which the system cannot provide any proposals remain empty.

   Using the *Context Menu (right mouse-button)* → *Entry Options,* or F4 Help, you select the InfoObjects that you want to assign to the DataSource fields. Alternatively, you can use the same data elements or field names to help you create an assignment.

   You do not have to assign InfoObjects to all the DataSources fields at this point. Using the transfer rules, you can also fill the InfoObjects of the communication structure with a constant or from a routine.

4. In the left half of the screen, the communication structure InfoObjects are displayed as well as the transfer rules that the system proposes.

   By selecting one row from both the left-hand side and the right-hand side of the screen, you can use the arrows to assign fields from the transfer structure to the InfoObjects of the communication structure.

You must remove from the transfer structure any fields that are not required. This improves performance, because otherwise data that you have not selected will be extracted.

5. For InfoObjects with the conversion routines ALPHA, NUMC or GJAHR, you can set the *Optional Conversion* key figure. See also Conversion Routines in BW [Page 98].
6. You can create a start routine [Page 283] if you use the PSA to load the data.

   This improves the system performance, for example, when you check if a certain request is already available in an ODS object, and makes the update rules consistent.
7. You can enhance or modify the transfer rules suggested by the system.

   To do this, select a transfer rule type by clicking on the corresponding *Type* symbol in the appropriate row:

   - **InfoObject:** The fields are transferred from the transfer structure and are not modified.

     Use the *Default transfer rules* function to assign fields in the transfer structure to fields in the communication structure.
   - **Constants:** An InfoObject is filled by a fixed value.

     You could, for example, assign the fixed value *US* to the InfoObject *0COUNTRY*.
   - **Formula**: An InfoObject is filled with a value that is determined using a formula [Page 236].
   - **Routine:** An InfoObject is filled from a local transfer routine.

     Local transfer routines are ABAP programs that you can create, modify, or transfer. The routine only affects the selected InfoObject in the relevant communication structure.

     For an explanation of the procedure see Creating Transfer Routines [Page 283].
8. Activate the transfer rules. Data can be loaded from the source system in an activated version only.

   The status of the transfer rules is shown as a green or a yellow traffic light.

   Since not all of the fields in the transfer structure have to be transferred into the communication structure, you can activate the transfer rules with just one assigned field. The status is shown as a yellow traffic light.

   A red traffic light indicates an error. The transfer rules cannot be activated if there are errors.

Result

You have ensured that the communication structure can be filled with data.

4.1.7.8.1.3.3.1 Start Routines in Transfer Rules

Use

You have the option of creating a start routine in the transfer rules maintenance screen. This start routine is run for each data package after the data has been written to the PSA and before the transfer rules have been executed. The entire data package in the transfer structure format is used as parameter for the routine.

Functions

You can change the data package by adding or deleting records.

If you add or delete records, this might not be detected by the error handling.

The start routine contains a return parameter that causes processing of the entire package to be terminated with an error message for values <> 0.

The option of creating a start routine is available only for the PSA transfer method. The routine is not displayed if you switch to the IDoc transfer method.

For general information on routines, see Update Routines and Start Routines [Page 314].

Example

You want to use an InfoSource with direct update to load additional texts from a flat file. You do not need the Japanese and Russian texts that are supplied with the file. These are filtered out by a start routine. The code for this start routine is shown below:

```
1.  *DATA:l_s_datapak_line type TRANSFER_STRUCTURE,
2.  * l_s_errorlog TYPE rssm_s_errorlog_int.
3.  delete datapak where LANGU='J'.
4.  delete datapak where LANGU='R'.
5.  *abort<>0 means skip whole data package!!
```

4.1.7.8.1.3.3.2 Creating Transfer Routines

Procedure

1. In the transfer rule maintenance screen, choose *Create Routine* for the relevant InfoObject.
2. For the transfer rule choose *Routine* → *Create* in the dialog box.
3. Specify a name for the local transfer routine that you want to create.
4. You have the option of using transfer structure fields in the routine. You can choose between
   - No fields:

The routine does not use any source structure fields. Make this selection when you determine the user name from a system variable (SY-UNAME), for example.

- All fields:

  The routine uses all source structure fields. In contrast to explicitly selecting all fields (see below), this option also includes fields that are added to the source structure later.

- Selected fields:

  If you make this selection, you have to explicitly select the fields used. Also in the program editor for implementing routines, only the selected fields are available to you in this case.

  You need these settings, for example when using SAP RemoteCubes, so that you can also determine the transfer structure fields for InfoObjects that are filled using transfer routines.

Choose *Next*. You get to the transfer routine ABAP editor.

5. Create a local transfer routine or change an existing routine.

   You can not delete the fields used in the routines from the transfer structure. They are displayed in the where-used list

   For SAP RemoteCubes you may have to create an inversion routine for transaction data. See also Inversion Routines [Page 284].

6. Save your entries.

See also:

Error Handling in Transfer Routines [Page 285]

4.1.7.8.1.3.3.3 Inversion Routine

Use

If you have defined transfer routines in the transfer rules for the InfoSource of a SAP RemoteCube, for performance reasons, it makes sense to also create inversion routines for each.

When jumping to a transaction in another SAP system using the report-report interface, you have to create an inversion routine for the transfer routine if you are using one, because otherwise the selections cannot be transferred to the source system.

Functions

You create an inversion routine in the routine editor for the already defined transfer routine. This routine is required, for example, during execution of queries on SAP RemoteCubes in order to transform the selection criteria for a navigation step into selection criteria for the extractor. The same goes for jumps to another SAP system with the report-report interface.

The form routine has the following parameters:

- I_RT_CHAVL_CS: The parameter contains the selection criteria for the characteristic in the form of a selection table.

- I_THX_SELECTION_CS: The parameter contains the selection criteria for all characteristics in the form of a hash table for selection tables of the individual characteristics. You only need this parameter if the inversion is still dependent on selection criteria of other characteristics.
- C_T_SELECTION: In this table parameter you have to return the transformed selection criteria. The table has the same structure as a selection table, but it also contains the field names in the FIELDNM component. If an empty table is returned for this parameter it means the table is a selection of all values for the fields used in the transfer routine. If an exact inversion is not possible, you can also return a superset of the exact selection criteria. In case of doubt, this is the selection of all values that was also provided as a suggestion during creation of a new transfer routine.
- E_EXACT: This key figures determines whether the transformation of selection criteria was executed exactly (constant RS_C_TRUE) or not (constant RS_C_FALSE).

Activities

Enter your program code for the inversion of the transfer routine between `*$*$ begin of inverse routine ...` und `*$*$ end of inverse routine ...` so that the variables C_T_SELECTION and E_EXACT are provided with the appropriate values.

With an inversion routine for a SAP RemoteCube it is sufficient if the value set is restricted in part. You do not need to make an exact selection.

With an inversion routine for a jump via RRI, you have to make an exact inversion so that the selections can be transferred precisely.

Example

You can find an example of the inversion routine by clicking *Routine Info* in the routine editor.

4.1.7.8.1.3.3.4 Error Handling in the Transfer Routine

In a transfer routine, you have the option of transferring error messages and warnings to the monitor.

Note the following:

- When you use the transfer routine to transfer messages to the monitor, you need to maintain in the scheduler the settings that control how the system behaves if an error occurs. See also Handling Data Records with Errors [Page 373].
- If, in your routine, you set the RETURNCODE <> 0, the record is transferred to error handling, but it is not posted.
- If, in your routine, you set the RETURNCODE = 0, the record is posted. If you transfer X-messages, A-messages, or E-messages to the monitor, the record is written to the error request at the same time, because the monitor table contains error messages.

  If you subsequently post this error request to the data target, records can be posted in duplicate. This does not happen if W-messages are transferred to the monitor.

4.1.7.8.1.4 Maintaining InfoSources (Flat File)

Purpose

You can load data from flat files (CSV or ASCII files) into the BI system.

You can upload the following data types:

1. Transaction data [Page 287]
2. Master data, either directly [Page 290] or flexibly [Page 287]
    - Attributes
    - Texts
3. Hierarchies [Page 291]

Prerequisites

Note the following with regard to CSV files:

- Excel files use separators to separate fields. In the European version, a semi-colon (;) is used as a separator. In the American version, a comma (,) is used. You can also use other separators. You must specify the delimiter used in the Scheduler.
- Fields that are not filled in a CSV file are filled with a blank space if they are character fields and with a zero (0) if they are numerical fields.
- If separators are used inconsistently in a CSV file, the “wrong” separator is read as a character, and both fields are merged into one field and possibly shortened. Subsequent fields are then no longer in the correct order.

Note the following with regard to CSV files and ASCII files:

- If your file contains headers that you do not want to be loaded, on the *External Data* tab page in the Scheduler, specify the number of headers that you want the system to ignore during the data load. This gives you the option of keeping the column headers in your file.
- A conversion routine determines whether or not you have to specify leading zeros. See also Conversion Routines in the BI System [Page 98].
- For dates, you usually use the format YYYYMMDD, without internal separators. Depending on the conversion routine, you can also use other formats.
- If you use IDocs to upload data, note the 1000 byte limit for each data record length. This limit does not apply to data that is uploaded using the PSA.

Notes on Uploading

- When you upload external data, you are able to load the data from any workstation into the BI system. However, from a performance point of view, you should store the data on an application server and load it from there into the BI system. This also means that you can load the data in the background.
- If you want to upload a large amount of transaction data from a flat file, and you are able to specify the file type of the flat file, you should create the flat file as an ASCII file. From a performance point of view, uploading the data from an ASCII file is the most cost-effective method. In certain circumstances, generating an ASCII file might involve a larger workload.

4.1.7.8.1.4.1 pdating Data Flexibly from a Flat File

Procedure

1. **Defining the Source System from Which You Want to Load Data**

   In the source system tree choose *File → Create.*

2. **Defining the InfoSource for Which You Want to Load Data**

   Optional: choose *InfoSource Tree → Root (InfoSources) → Create Application Components.*

   Choose *InfoSource Tree → Your Application Component → Additional Functions → Create InfoSource 3.x → Flexible Updating.* Enter a name and a description

3. **Maintaining the Communication Structure: defining the Fields for the Flat Files as InfoObjects in the BI System**

   Specify an InfoObject for each column of your flat file. You can either use existing InfoObjects or create new ones.

   More information:

   Creating InfoObjects: Characteristics [Page 60]

   Creating InfoObjects: Key Figures [Page 107]

   The sequence of columns in your communication structure does **not** have to correspond to the sequence of columns in your flat file.

   Activate the communication structure.

4. **Assigning the Source System to the InfoSource**

   Expand the *Transfer Structure/Transfer Rules* in the lower half of your screen and select your source system.

   A proposal for the DataSource, the transfer structure, and the transfer rules is generated automatically.

5. **Maintaining the Transfer Structure/Transfer Rules**

   Change the transfer structure or the transfer rules where necessary.

   More information: InfoSources with Flexible Updating of Flat Files [Page 288]

   The sequence of columns in the transfer structure must correspond to the sequence of columns in your flat file. If you do not use the same sequence, the corresponding transfer structure is filled incorrectly.

   Activate the transfer structure/transfer rules.

Further Steps:

For example, InfoCube:

Creating InfoCubes [Page 130]

Creating Update Rules for InfoProviders [Page 304]

Maintaining InfoPackages [Page 343]

Checking the Data Loaded in the InfoCube [Page 136]

4.1.7.8.1.4.1.1 InfoSource with Flexible Update for Flat Files

Purpose

If you want to load data from a flat file into BW, you have to maintain the relevant transfer structure and transfer rules in BW manually. There is no function for automatically uploading Metadata.

You can use flexible updating for transaction data and master data in any kind of data target except hierarchies (InfoCubes, ODS Objects, InfoObjects).

Process Flow

In the transfer structure maintenance, specify an InfoObject for every field of your flat file, making sure that the **sequence** of the InfoObjects corresponds to the sequence of the columns in your flat file. If you do not use the same sequence, the corresponding transfer structure is not filled correctly.

For the flat file structure,

```
19980101;0001;23
```

the corresponding transfer structure could be:

0CALDAY
PRONR
PROPRICE

0CALDAY describes the date (01.01.1998) as an SAP time-characteristic, PRONR describes the product number (0001) as the characteristic, and PROPRICE describes the product price as the key figure.

Specify the **data types** according to the fields that you want to upload from the flat file.

If the data for your flat file was staged from an SAP system, there are no problems when transferring data types into BI. Please note that you might not be able to load the data types DEC and QUAN for flat files with external data. Specify type CHAR for these data types in the transfer structure. When you load, these are then converted into the data type, which you specified in the maintenance of the relevant InfoObject in BW.

If you want to load an exchange rate from a flat file, the format must correspond to the table TCURR.

You have to select a suitable **update mode** in transfer structure maintenance so that the system uses the correct update type.

- Full upload (ODS Object, InfoCube, InfoObject)

  The DataSource does not support delta updates. With this procedure, a file is always copied in its entirety. You can use this procedure for ODS objects, InfoCubes and also InfoObjects.

- Latest status of changed records (ODS objects only)

The DataSource supports both full updates and delta updates. Every record to be loaded defines the new status for all key figures and characteristics. This procedure should only be used when you load into ODS objects.

- Additive delta (ODS object and InfoCube)

  The DataSource supports both full updates and additive delta updates. The record to be loaded only provides the change in the key figure for key figures that can be added. You can use this procedure for ODS objects and for InfoCubes.

  Example of loading flat files:

  The customer orders 100001 and 100002 are transferred to BW with a delta initialization.

  Delta initialization:

Document No.	Document Item	...	Order Quantity	Unit of Measure	...
100001	10		200	Pieces	
100001	20		150	Pieces	
100002	10		250	Kg	

  After delta initialization, the order quantity of the first item in customer order 100001 is reduced by 10% and the order quantity of the second item increased by 10%. There are then two options for the file upload of the delta in an ODS Object.

  1. Option: Delta process shows the latest status for modified records (applies to ODS Object only):

Document No.	Document Item	...	Order Quantity	Unit of Measure	...
100001	10		180	Pieces	
100001	20		165	Pieces	

  CSV file:

  100001;10;...;180;PCS;...

  100001;20;...;165;PCS;...

  2. Option: Delta process shows the additive delta (applies only to InfoCube/ODS object):

Document No.	Document Item	...	Order Quantity	Unit of Measure	...
100001	10		-20	Pieces	
100001	20		15	Pieces	

  CSV file:

  100001;10;...;-20;PCS;...

  100001;20;...;+15;PCS;...

To make sure that the data is uploaded in the correct structure, you can look at it in the preview and simulate the upload. See Preview and Simulation of Loading Data from Flat Files [Page 299].

Result

You have maintained the metadata for the InfoSource with flexible update and can now upload the data from the flat file.

4.1.7.8.1.4.2 pdating Master Data from a Flat File

Procedure

1. **Defining the source system from which you want to load data**

   In the source system tree choose *File* → *Create.*

2. **Defining the InfoSource for which you want to load data**

   Optional: Choose *InfoSource Tree* → *Root (InfoSources)* → *Create Application Components.*

   Choose *InfoSource Tree* → *Your Application Component* → *Other Functions* → *Create InfoSource 3.x* → *Direct Update of Master Data.*

   Choose an InfoObject from the proposal list, and specify a name and a description.

3. **Assigning the source system to the InfoSource**

   Choose *InfoSource Tree* → *Your Application Component* → *Your InfoSources* → *Assign Source System*.

   You are taken automatically to the transfer structure maintenance.

   The system automatically generates DataSources for the three different data types to which you can load data.

   1. Attributes
   2. Texts
   3. Hierarchies (if the InfoObject has access to hierarchies)

   The system automatically generates the transfer structure, the transfer rules, and the communication structure (for attributes and texts).

4. **Maintaining the transfer structure / transfer rules**

   Choose either the DataSource to load attributes or the DataSource to load texts.

   The system automatically generates a proposal for the data source, transfer structure, transfer rules and communication structure for you.

   Attributes

   The proposal for uploading attributes displays which structure your flat file must have for uploading attributes, and contains at least the characteristic and the attributes assigned to it. Make sure that the sequence of the objects in the transfer structure corresponds to the sequence of the fields in the flat file.

   The following fields can be required in a flat file for attributes:

/BIC/<ZYYYYY>		Key for the compounded characteristic (if the characteristic exists)

/BIC/<ZXXXXX>		Characteristic key
DATETO	CHAR 8	valid to – date (only for time-dependent master data)
DATEFROM	CHAR 8	valid from – date (only for time-dependent master data)

Texts

The proposal for uploading texts displays which structure your flat file must have for uploading texts for this characteristic. Ensure that the structure of your flat file corresponds to the proposed structure.

The following fields can be required in a flat file for texts:

LANGU	CHAR 1	Language key (F for French, E for English)
/BIC/<ZYYYYY>		Key for the compounded characteristic (if the characteristic exists)
/BIC/<ZXXXXX>		Characteristic key
DATETO	CHAR 8	valid to – date (only for time-dependent master data)
DATEFROM	CHAR 8	valid from – date (only for time-dependent master data)
TXTSH	CHAR 20	Short text
TXTMD	CHAR 40	Medium-length text
TXTLG	CHAR 60	Long text

The sequence of columns in the transfer structure must correspond to the sequence of columns in your flat file. If you do not use the same sequence, the corresponding transfer structure is not filled correctly.

Activate the transfer structure/transfer rules and the communication structure.

Further Steps:

Maintain InfoPackage [Page 343]

4.1.7.8.1.4.3 ploading Hierarchies from Flat Files

Prerequisites

If you want to load InfoObjects in the form of hierarchies, you have to activate the indicator *with hierarchies* for each of the relevant InfoObjects in the InfoObject maintenance [Page 83]. If necessary, you need to specify whether the whole hierarchy or the hierarchy structure is to be time-dependent, whether the hierarchy can contain intervals, whether additional node attributes are allowed (only when loading using a PSA), and which characteristics are allowed.

Procedure

1. **Defining the source system from which to load data**

Choose the source system tree *File → Create.*

2. **Defining the InfoSource for which you want to load data**

   Optional: choose *InfoSource Tree → Root (InfoSources) → Create Application Components.*

   Choose *InfoSource Tree → Your Application Component → Other Functions → Create InfoSource 3.x → Direct Update.*

   Choose an InfoObject from the proposal list, and specify a name and a description.

3. **Assigning the source system to the InfoSource**

   Choose *InfoSource Tree → Your Application Component → Your InfoSource → Assign Source System*. The transfer structure maintenance screen appears.

   The system automatically generates DataSources for the three different data types to which you can load data.

   - Attributes
   - Texts
   - Hierarchies (if the InfoObject has access to hierarchies)

   The system automatically generates the transfer structure, the transfer rules, and the communication structure (for attributes and texts).

4. **Maintaining the transfer structure / transfer rules**

   Select the DataSource for uploading hierarchies.

   IDoc transfer method: The system automatically generates a proposal for the DataSource and the transfer structure. This consists of an entry for the InfoObject for which hierarchies are loaded. With this transfer method, the structure is converted to the structure of the PSA during loading, which affects performance.

   PSA transfer method: The transfer methods and the communication structure are also generated here.

5. **Maintaining the hierarchy**

   Choose *Hierarchy Maintenance,* and specify a technical name and a description of the hierarchy.

   You load a sorted hierarchy **without** expanded leaf values and node InfoObjects. The system generates a table with the following hierarchy format:

Description	Field Name	Length	Type
Node ID	NODEID	8	NUMC
InfoObject name	INFOOBJECT	30	CHAR
Node name	NODENAME	32	CHAR
Catalog ID	LINK	1	CHAR
Higher-level node	PARENTID	8	NUMC
First subnode	CHILDID	8	NUMC
Next adjacent node	NEXTID	8	NUMC
Language key	LANGU	1	CHAR
Description - short	TXTSH	20	CHAR

Description - medium	TXTMD	40	CHAR
Description - long	TXTLG	60	CHAR

We recommend that you use the **transfer method PSA**: Here you can set the indicator *Expand Leaf Values and Node InfoObjects*. As a result, characteristic values are not transferred to the hierarchy fields NODENAME, LEAFFROM and LEAFTO as is normally the case, but to their own transfer structure fields. With this option you can load characteristic values whose node name length corresponds to the maximum length possible for characteristics (currently 60). Prerequisite is that you activated the option *Expanded leaf values*.

The system transfers the settings for the intervals and for time-dependency from the InfoObject maintenance. Depending on which settings you have defined in the InfoObject maintenance, further table fields can be generated from the system.

The *valid from* and *valid to* field is filled if you select *Total Hierarchy Time-dependent* in the InfoObject maintenance. The *time-dependent* indicator is activated if you select the *Hierarchy Nodes Time-dependent* option in the InfoObject maintenance.

6. **Save your entries.**

Depending on which settings you defined in the InfoObject maintenance, additional fields can be generated by the system. Also note the detailed description for Structure of a Flat Hierarchy File for Loading via an IDoc [Page 293] and for Structure of a Flat Hierarchy File for Loading Using a PSA [Page 296].

Further Steps:

Maintaining InfoPackages [Page 343]

4.1.7.8.1.4.3.1 tructure of a Flat Hierarchy File for Loading via IDoc

Prerequisites

You have created a sorted hierarchy and the system generates a structure with the following hierarchy format:

Description	Field name	Length	Type
Node ID	NODEID	8	NUMC
Name of the basic characteristic of the hierarchy	INFOOBJECT	30	CHAR
Node name	NODENAME	32	CHAR

Catalog ID	LINK	1	CHAR
Higher-level node	PARENTID	8	NUMC
First subnode	CHILDID(*)	8	NUMC
Next adjacent node	NEXTID(*)	8	NUMC
	DATETO*	8	CHAR
	DATEFROM*	8	CHAR
	LEAFTO*	32	CHAR
	LEAFFROM*	32	CHAR
Language key	LANGU	1	CHAR
Description - short	TXTSH	20	CHAR
Description - medium	TXTMD	40	CHAR
Description- long	TXTLG	60	CHAR

- The rows marked in green (*) are only generated automatically if a sorted hierarchy is being used.
- The rows marked blue* are only automatically generated if you have created an InfoObject with a time-dependent hierarchy and/or intervals.

Process Flow

The following table shows which data you should upload from the flat file.

NODEID	NUMC 8	Specify the internal ID of the hierarchy node.
INFOOBJECT	CHAR 30	In the row INFOOBJECT, specify the name of the basic characteristic to which the hierarchy should be assigned. The master data for this characteristic is then used. Alternatively, you can include a text node. Text nodes are always based on InfoObject 0HIERNODE. You can use text nodes if you need country or city names for the evaluation criteria of a hierarchy.
NODENAME	CHAR 32	For master data, enter the key of the master data table. Enter any name you choose for text nodes.
LINK	CHAR 1	With ‘normal’ nodes, leave the field empty. If the node is a link node [External] that is, if the node is a lower-level node with two higher-level nodes, create two rows for the InfoObject. Then create a row and leave the field LINK empty. In the second row, create the InfoObject as the lower-level of the second higher-level node with a new NODEID, but give it the same NODENAME. Also enter an ‘X’ in the LINK column. If you enter the ‘X’, a link exists between this node and the second node by the same name. This means that the node has the same subtree as the second node. If you change the structure of the second node, the structure of the link node also changes.
PARENTID	NUMC 8	Enter the NODEID for the first higher-level node. Enter, “00000000“ if

		there is no higher-level node.
CHILDID	NUMC 8	Enter the NODEID for the first lower-level node. Enter, “00000000“ if there is no lower-level node.
NEXTID	NUMC 8	Enter the NODEID for the first ‘next node’. Enter, “00000000“ if there is no ‘next node’.
DATETO	CHAR 8	Valid–to nodes (are needed if the hierarchy structure is time-dependent).
DATEFROM	CHAR 8	Valid–to nodes (are needed if the hierarchy structure is time-dependent).
LEAFTO*	CHAR 32	Upper limit of a hierarchy interval (needed if the hierarchy contains intervals).
LEAFFROM*	CHAR 32	Upper limit of a hierarchy interval (needed if the hierarchy contains intervals).
LANGU	CHAR 8	Enter the language ID (is required for text nodes) For example, F for French, E for English, and so on.
TXTSH	CHAR 8	Enter a short text. This is needed for text nodes, as no texts can be loaded for these nodes.
TXTMD	CHAR 32	Enter a medium text. This is needed for text nodes, as no texts can be loaded for these nodes.
TXTLG	CHAR 32	Enter a long text. This is needed for text nodes, as no texts can be loaded for these nodes.

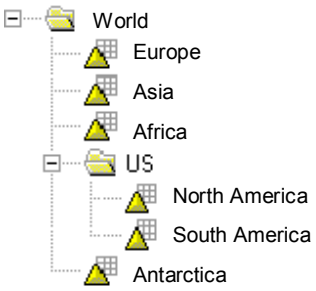
Result

You have created a flat file with a hierarchy, which you can save as a CSV or ASCII file and upload into the BW as an IDoc.

Structuring a Simple Hierarchy File - Example:

NODEID	INFOOBJECT	NODENAME	LINK	PARENTID	CHILDID	NEXTID	LANGU	TXTSH
00000001	0HIER_NODE	EA			00000002		D	World
00000002	MTCONT	EU		00000001		00000003		
00000003	MTCONT	AS		00000001		00000004		
00000004	MTCONT	AF		00000001		00000005		
00000005	0HIER_NODE	US		00000001	00000006	00000008		
00000006	MTCONT	NA		00000005		00000007		
00000007	MTCONT	SA		00000005				
00000008	MTCONT	AA		00000001				

The file displayed above would correspond to the following hierarchy:



The names of the characteristic nodes (yellow triangles) are taken from the master data text table. The names of the text nodes (folders) are taken from the hierarchy table above. In this example, only the name ‘World’ was entered in the hierarchy table. This is why the NODENAME ‘US’ is displayed for the second text node.

4.1.7.8.1.4.3.2 Structure of a Flat Hierarchy File for Loading Using a PSA

Prerequisites

You have created a sorted hierarchy and the system generates a structure with the following hierarchy format. The hierarchy basis characteristic has n node attributes, is compounded m times, and has additional node characteristics that contain additional fields with compounding l:

Description	Field Name	Length	Type
Node ID	NODEID	8	NUMC
InfoObject name	INFOOBJECT	30	CHAR
Node name	NODENAME	32	CHAR
Catalog ID	LINK	1	CHAR
Parent node	PARENTID	8	NUMC
First subnode	CHILDID(*)	8	NUMC
Next adjacent node	NEXTID(*)	8	NUMC
	DATETO*	8	CHAR
	DATEFROM*	8	CHAR
	LEAFTO*	32	CHAR
	LEAFFROM*	32	CHAR
Language key	LANGU	1	CHAR
Description - short	TXTSH	20	CHAR
Description - medium	TXTMD	40	CHAR
Description- long	TXTLG	60	CHAR

<Node attribute 1>			
...			
<Node attribute n>			
<Node field 1 >*			
...			
<Node field m >*			
<Node field m+1 >			
...			
<Node field m+1>			
<To field 1>*	TO-**		
...	TO-**		
<To field m>*	TO-**		

- The rows marked in green(*) are only generated automatically if a sorted hierarchy is being used.
- The rows marked blue* are only automatically generated if you have created an InfoObject with a time-dependent hierarchy and/or intervals.
- The rows marked red are only automatically generated if you have permitted additional node attributes.
- The rows marked red* are only automatically generated if you have set the *Expand Leaf Values and Node InfoObjects* indicator in the maintenance of the hierarchy header in the InfoSource maintenance. The to- fields are inserted with the same name as the from-fields, but in their own sub-structure (TO-**).

Choosing the *Expand Leaf Values and Node InfoObjects* pushbutton displays the hierarchy structure. For additional details about this function, also refer to the **Hierarchy Maintenance** section of Uploading Hierarchies from Flat Files [Page 291].

Process

You can find out how you need to fill the fields in the section **Process** in Structure of a Flat Hierarchy File for Loading Using an IDoc [Page 293].

Result

You have created a flat file with a hierarchy, which you must save as a CSV or ASCII file. You can then upload it into the BW using the PSA.

Structuring a Simple Hierarchy File - Example:

Without expanded leaf values and node InfoObjects:

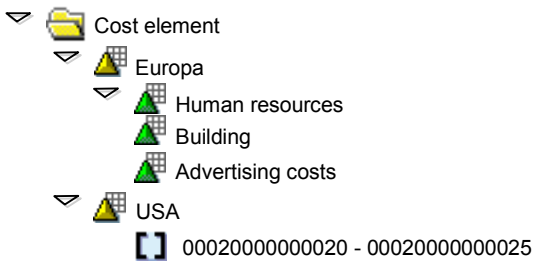
NODE-ID	IOBJNM	NODENAME	L	PARENT-ID	CHILD-ID	NEXT-ID	LEAF TO	LEAF FROM	L	TXTSH	...	SIGNCH
1	0HIER_NODE	COSTELMNT		0	2	0			D	Costelmnt		
2	0CO_AREA	1		1	4	3						
3	0CO_AREA	2		1	7	0						
4	0COSTELMNT	11		2	0	5						
5	0COSTELMNT	12		2	0	6						
6	0COSTELMNT	13		2	0	7						
7	0COSTELMNT			3	0	0	20	25				X

With expanded leaf values and node InfoObjects:

Here the columns controlling area (CO_AREA) and cost element (COSTELMNT), as well as interval controlling area (CO_AREA) and interval cost element (COSTELMNT) are added.

NODE-ID	IOBJNM	NODENAME	L	PARENT-ID	CHILD-ID	NEXT-ID	L	TXTSH	...	SIGNCH	CO_AREA	COSTELMNT	CO_AREA	COST
1	0HIER_NODE	COSTELMNT		0	2	0	D	Costelmt						
2	0CO_AREA			1	4	3					1			
3	0CO_AREA			1	7	0					2			
4	0COSTELMNT			2	0	5						11		
5	0COSTELMNT			2	0	6						12		
6	0COSTELMNT			2	0	7						13		
7	0COSTELMNT			3	0	0				X	2	20	2	

The files displayed above would correspond to the following hierarchy:



This hierarchy of cost elements has the following properties:

- It has 0COSTELMNT as a hierarchy basic characteristic, which is compounded to 0CO_AREA. That is why both characteristics have to be included in the file. For expanded leaf values and node InfoObjects both characteristics have to be included twice for intervals: An interval is assigned to the USA node, which includes the cost element 20-25. The file needs the fields 0CO_AREA and 0COSTELMNT twice for the interval, for both the from- and to- value. The interval description is made up from a

combination of the specifications for these two characteristics: 0002 for 0CO_AREA and 0000000020 for 0COSTELMNT.

- The controlling area (0CO_AREA) has to be maintained as an external characteristic for the cost element InfoObject (0COSTELMNT).
- The hierarchy is a sorted hierarchy.
- This node also has the node attribute sign change, meaning that the cost element can be displayed as a negative in the query. For this reason, an X is uploaded for this node representing the sign change (SIGNCH)

See also:

Special Features when Loading via the PSA [External]

4.1.7.8.1.4.4 Preview and Simulation of Loading Data from Flat Files

Use

Before you load data from a flat file, you can take look at the data in the preview. This lets you check the data before you load it.

From the preview, you can run a simulation of the data loading process. This allows you to check the update process.

This function makes it easier for you to check that the structure of the CSV and ASCII files you want to load is correct. It provides you with a better overview of data, particularly with hierarchies.

Prerequisites

You have created and activated the transfer structure of the InfoSource. To simulate the upload, you also have to create and activate update rules.

Features

Preview

Once you have selected the file parameter information, the transfer structure is displayed in the preview as it appears after loading.

Simulation

The data loading process is simulated. Note that only the PSA transfer method is supported. With transaction data, the transfer rules and the update rules are simulated, and you can look at the filled communication structure or the updated InfoCube. With attributes and texts, the transfer rules are simulated, allowing you to take a look at the filled communication structure. With hierarchies, the hierarchy tree is displayed along with any error messages.

Activities

1. Choose the *Preview* pushbutton from the transfer structure maintenance of your InfoSource.
2. Choose the file parameters for the preview, and then choose *Execute*. You get a preview of the transfer structure for your file after the upload.

3. From the preview, you are then able to use the *Simulate* pushbutton to simulate the upload of the file. If you want to simulate transaction data that can be updated in several InfoCubes or DataStore objects, you can select the InfoProvider. With attributes, texts, and hierarchies, the simulation is simply started.

4.1.7.8.1.5 Maintaining InfoSources (External System)

You can also assign an external system to an InfoSource. Data is then loaded from this system into the BI system using staging BAPIs. There are a number of extraction tools from third-party providers that you can use to load data from an external system, or you can use the DB connect tool provided by BI. You maintain a destination for an extraction tool; this destination is accessed when you load data from the BI system. The extraction tool loads the data from the external system and transforms it into the relevant BI format.

When you load transaction data from an external system, you must make sure that the transfer structure corresponds exactly to the data structure of the extraction tool.

When you load master data from an external system, you must make sure that the data structure of the extraction tool corresponds exactly to the transfer structure suggested by the BI system.

Procedure

1. **Defining the Source System from Which You Want to Load Data**

   In the source system tree, choose *External System* → *Create.*

   Enter a name and a description and maintain the RFC destination for your extraction tool.

2. **Defining the InfoSource for Which You Want to Load Data**

   The InfoSource maintenance and subsequent processes are identical to when you load data from a flat file. Choose a procedure according to data type:

   - Uploading Data Flexibly from a Flat File [Page 287]
   - Loading master data from flat files directlysapu [Page 290] or flexibly [Page 287]
   - Uploading Hierarchies from Flat Files [Page 291]

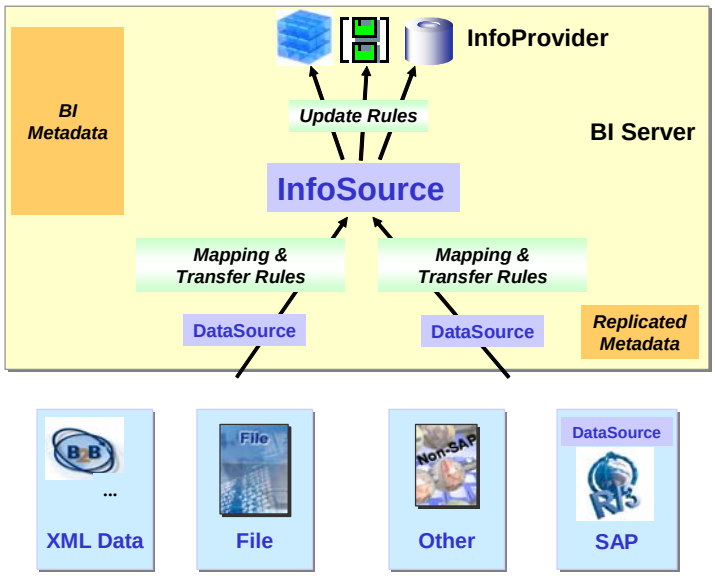
4.1.7.8.2 Update Rules

Definition

Update rules specify how data (key figures, time characteristics, characteristics) is updated into the InfoProvider from the communication structure of an InfoSource. You are therefore connecting an InfoSource with an InfoProvider.

As of SAP NetWeaver 2004s, the concept of update rules has been replaced by the new transformation concept. You can still create update rules, but we recommend that you use the new transformation concept. Transformations provide an easy-to-use, graphical maintenance interface. There is no longer a distinction between transfer and update rules; you only need to create transformation rules.

In the Data Warehousing Workbench, the ▫ symbol before the description identifies an object that is available for the new concept.



Use

An update rule must be specified for each key figure and the corresponding characteristics of the InfoCube. For a DataStore object, it must be specified for the data and key fields; for an InfoObject it must be specified for the attribute and key fields.

The following update types exist:

1. No Update
2. Addition, Minimum, or Maximum
3. Overwrite (DataStore objects and InfoObjects only)

A data target can be filled with data from several InfoSources. A set of update rules must be maintained for each of these InfoSources. These describe how the data is to be written into the data target from the communication structure for the InfoSource.

In the update, the InfoObjects of the communication structure for the InfoSource are referred to as source InfoObjects (source key figure or source characteristic). The InfoObjects for the InfoCube, on the other hand, are referred to as target InfoObjects (target key figure or target characteristic). Similarly, the InfoObjects of a DataStore object or InfoObject are referred to as target InfoObjects (target data field or target key field).

Structure

There is an update rule for each key figure of the InfoSource. This consists of the rule for the key figure itself and the rules for the characteristics, time characteristics, and units assigned to the key figure.

4.1.7.8.2.1 Update Options

Use

1. **Each characteristic of the InfoProvider also exists in the InfoSource.**

Data Record 1 of the InfoSource

CHW	*Boston*	*January*	*1000*	*1000*

Data Record 2 of the InfoSource

CHW	*Boston*	*January*	*1000*	*1000*

InfoCube

Characteristics			Key Figures	
Plant	Storage	Month	Returns 1	Returns 2
CHW	*Boston*	*January*	*1000*	*1000*

Data records 1 and 2 of the InfoSource have the same characteristic values (plant, storage location and month). The key figures of these two data records are transferred into the InfoCube according to the relevant update rule. This example uses adding.

2. **A characteristic of the InfoProvider does not appear in the InfoSource.**

   If this situation should arise, you have two options open to you:

   a. Assign a suitable characteristic for the InfoSource with the same type (characteristic, time characteristic or unit characteristic) to the characteristic for the InfoCube.

Data Record 1 of the InfoSource

Characteristics			Key Figures	
Plant	**Storage**	Month	Returns 1	Returns 2
CHW	*Boston*	*January*	*1000*	*1000*

InfoCube

Characteristics			Key Figures	
Plant	**Storage**	Month	Returns 1	Returns 2
CHW	*Boston*	*January*	*2000*	*2000*

The characteristic *Location* in data record 1 of the InfoSource is assigned to the characteristic of the same type *Storage Location* in the InfoCube.

b. Write an update routine that calculates the corresponding characteristic of the InfoCube.

c. Enter a value (constant) that you want the characteristic of the InfoCube to have.

d. Fill the characteristic by reading the master data for another characteristic of the communication structure.

e. Do not enter anything for the characteristic and leave it blank.

3. **A characteristic exists in the InfoSource, but does not appear in the InfoProvider.**

   In this case, the corresponding characteristic is not updated.

4.1.7.8.2.2 Update Process

Purpose

In order to better understand the update rules maintenance, make yourself familiar with the underlying update mechanism.

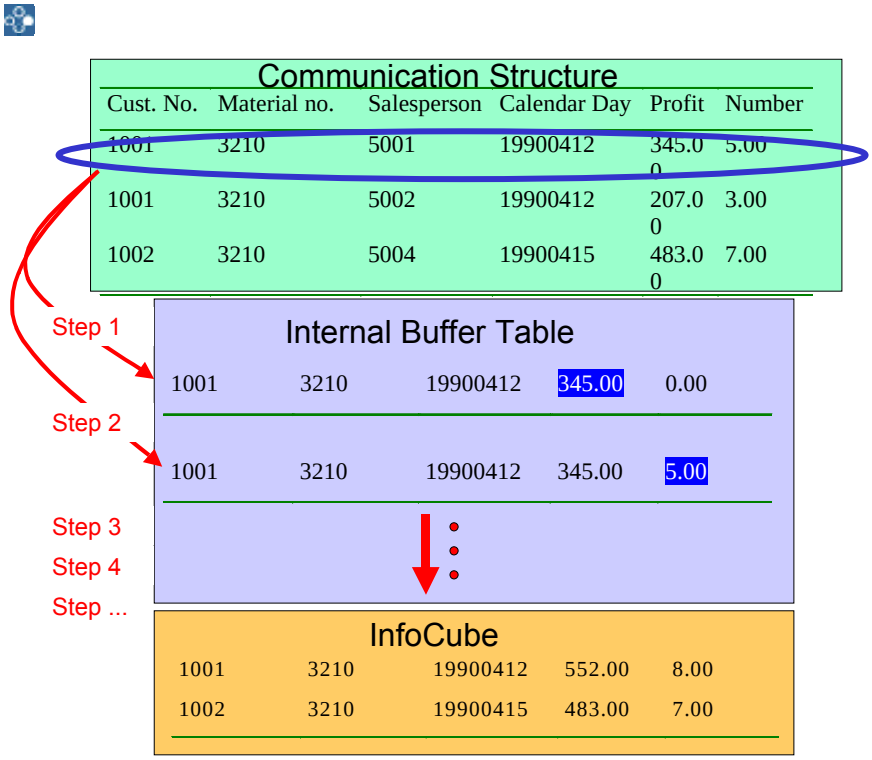
Process

You load a request into the InfoCube (or into any other InfoProvider). For each existing key figure loaded into the InfoCube, the update mechanism calculates the characteristic values and puts the results into an intermediate buffer.

If more than one key figure is updated, each individual entry is then processed for a key figure, meaning that the value of the key figure is calculated initially and then all values of the characteristics. The values of other key figures are initially set to zero. All entries are thus processed specific to key figures one after another.

Entries that are identical in characteristic values, are aggregated for an entry in the buffer table. With the aid of the update routines, you do have the option, however, of using the function Splitting Key Figures [Page 310] and, thereby, generating more than one entry from a single entry.

The processed entries are updated in the data target by the internal table.



In the example graphic above, three records from the communication structure are updated that are processed one after another. The characteristic *sales person* does not exist in the InfoCube and is, therefore, not updated.

The characteristic values for the first key figure (profit) of the first record is calculated initially. The key figure value of the second key figure (number) of the first record is first set to zero.

In the next step, the characteristic values are calculated for the second key figure of the first record. Since the characteristic values for both key figures concur, they are aggregated.

The other records are processed in this way, one after the other. Since the characteristic *sales person* is not updated, the characteristic values for both key figures are the same in the first two records. These are consequently aggregated up to one record.

4.1.7.8.2.3 Creating Update Rules for InfoProviders

Prerequisites

To maintain update rules, choose *Modeling* → *InfoProvider* → *Context Menu for Your InfoProvider* → *Additional Functions* → *Create Update Rules* in the Data Warehousing Workbench.

Procedure

1. Select a data source. Choose between InfoSources, DataStore objects and InfoCubes. You can select a template for the update rule.
2. Choose *Next Screen* .

   A proposal for the update rule is generated from the InfoProvider. To display the technical name used to describe the fields, choose *Technical Name* .
3. Select a field by double-clicking on it, or choose *Type*.
4. Define the update type [Page 305] for the displayed key figure or data field.
5. Select an update method [Page 307].
6. Also select an update method for the characteristics and the time reference or for the key fields.

   Note: For non-cumulative InfoCubes, only the time reference characteristic and fiscal year variant can be maintained. All other time characteristics are automatically derived from the time-reference characteristic. Therefore, the time-reference characteristic must not be left blank.
7. Select the next key figure by double clicking on it, or by selecting *Type,* and repeat steps 4 to 6.
8. Choose *Save* if you want to save the rules without the system performing an update.

   Choose *Activate* if you want the system to perform an update. An InfoProvider has to have an active update rule before it can contain data.

4.1.7.8.2.3.1 Update Types

Use

With the update type, you control whether a key figure/data field is updated in the InfoProvider.

Functions

For InfoCubes:

- Depending on the aggregation type you entered in the Key Figure Maintenance [Page 107] for this key figure, you are given the options ***Addition*** or ***Maximum*** or ***Minimum***. If you choose one of these options, new values are updated in the InfoCube.

  The **aggregation type** (**addition, minimum & maximum**) determines how key figures are updated where primary keys are the same. Thus the total, the minimum or the maximum for these values is formed for new values.

  If you entered one of these aggregation types in the key figure definition, it is transferred for the update. Otherwise, the aggregation type addition is automatically selected.

- If you choose ***no update***, the key figures are not updated in the InfoCube, meaning that no data records are written in the InfoCube with the first data transfer, or that data records that already exist, will remain in place with subsequent transfers.

For InfoObjects:

Only the ***Overwrite*** option is available. With this option, new values are updated to the InfoObject.

For ODS Objects:

1. Depending on the type of data and DataSource, you have the options ***Addition, Minimum, Maximum*** or ***Overwrite***. If you choose one of these options, new values are updated in the ODS object.

   For numerical data fields, you are given a proposal for the update type through the characteristic 0RECORDMODE. If only the after-image is delivered, the system proposes *Overwrite*. However, it could make sense to change this: For example, with the meter data field "# Changes", which is filled with a constant 1 but still has to be updated through addition, although only an after-image is delivered.

   Characteristic 0RECORDMODE is used to pass indicators of the DataSource (from SAP systems) to the update.

   You do not need characteristic 0RECORDMODE as long as you do not load delta requests in the ODS object or only from file DataSources.

   a. ***Addition***:

      Addition is not supported with the data types CHAR, DAT, TIMS, CUKY and UNIT. To be able to use the addition function, make sure the DataSource is compatible with an additive delta.

   b. ***Overwrite***:

      Whether overwriting is supported is determined by the delta-compatibility of the DataSource.

      In this example, the order quantity changes after it has been loaded into BW. With the second load process the data is overwritten since it has the same primary key.

      First Load Process

Document No.	Document Item	Order Quantity	Unit of measure
100001	10	200	Pieces
100001	20	150	Pieces
100002	10	250	Kg

      Second Load Process

Document No.	Document Item	Order Quantity	Unit of measure
100001	10	180	Pieces

100001	20	165	Pieces

When you update data, the system keeps to the time sequence of the data packages and requests. You have to decide the logical order of the update yourself. Meaning, for example, orders must be requested before deliveries; otherwise the wrong results may appear when you overwrite the data.

c. If you choose ***no update***, the data fields are not updated in the ODS object, meaning that no data records are written in the ODS object with the first data transfer, or that data records that already exist will remain in place with subsequent transfers.

4.1.7.8.2.3.2 pdate Methods

Use

You use the calculation method to control whether and how a characteristic/key figure or a data field/key field should be updated in the data target.

Functions

There are several options:

- **Source InfoObject**: The field is filled directly from the selected source InfoObject of the communications structure.

  If the system does not make any suggestions for a *source InfoObject*, you can assign a *source InfoObject* of the same type (amount, number, integer, quantity, float, time) or create a *routine*.

  If you assign a source InfoObject **of the same type**, that has a different currency to the target InfoObject, then you must translate the source currency using a Currency Translation to the Target Currency [Page 312].

  For source-time characteristics: The system offers Automatic Time Conversion [Page 308].

- **Constant**: The field is not filled with the communication structure. It is filled directly with the specified value.

- **Formula**: The key figure/data field/attribute is updated with a value determined with a Formula [Page 236].

- **Master Data Attribute of**: The data field/attribute is updated by reading the master data table of an InfoObject that is included with a key and a value in the communication structure and contains the corresponding data field/attribute as an attribute. The attributes and their values are read from the key, these are then returned.

  In an InfoCube there is a characteristic (for example, FM area) that does not appear as a characteristic in the communication structure. In the communication structure, however, there is a characteristic (for example, cost center) that has the characteristic *FM area* as an attribute.

  You can read the attribute *FM area* from the master data on demand, and thereby fill the characteristic *FM area* in the InfoCube.

It is not possible to read recursively, that is, to read additional attributes for the attribute To do this, you have to use routines.

If you have changed master data, you have to execute the change run [Page 689]. By reading the master data, the active version is read. If this is not available, an error occurs.

If the attribute is time-dependent, you also have to define when it should be read: at the current date (sy-datum), at the beginning or end of a period (defined with a time characteristic of the InfoSource), or at a constant date that you enter directly. *sy-datum* is used as a default.

- **Routine**: The field is filled by an Update Routine [Page 317] that you have written.

  The system provides you with a selection option which lets you decide whether the routine should be valid for all of the key figures /data field / attributes belonging to this characteristic or only for the key figure / data field / attribute displayed.

  Update routines generally only have one return value. If you select *return table*, the corresponding key figure routine then no longer has a return value, but a return table. You can then generate as many key figures/data field values as you like from one data record. [Page 310]

  With ODS objects/InfoObjects: You cannot use the return code in the routine for data fields that are updated by being overwritten. If you do not want to update specific records, you can delete these from the start routine.

  If you create different rules for different key figures/data fields for the same characteristic [Page 311], a separate data record can be created from a data record of the InfoSource for each key figure.

  For InfoCubes: If you choose a routine, you can then also choose the indicator *routine with unit calculation*. In the routine you then also get the return parameter ‘UNIT’. In this respect you can, for example, store the required unit of the key figure, such as ‘DEM’ or ‘ST’. You can use this option, for example, to calculate the unit KG present in the communication structure in tons in the InfoCube

  If you fill the target key figure from an update routine for update rules, the currency translation has to be carried out using the update routine. This means that an automatic calculation is not available.

- **Initial value**: The field is not filled. It remains empty.

4.1.7.8.2.3.2.1 Automatic Time Conversion

Use

In the update rule maintenance, you can use automatic time conversion to update InfoSource source time characteristics into target time characteristics in the data target. This function does not apply to ODS objects, since time characteristics are treated like normal data fields.

Prerequisites

In the update rule maintenance, you have already determined rules for the corresponding key figure and associated characteristic. You are now in the *Time Reference* tabpage for InfoCubes, and in the *Key Fields* tab page for ODS objects.

Functions

The system only makes those time characteristics available, for which an automatic time conversion routine exists.

For the target time characteristic *Calendar Month,* the source time characteristic *Calendar Day* is made available. This means that a time conversion routine exists, that converts the source time characteristic *Calendar Day* into the target time characteristic C*alendar Month*.

4.1.7.8.2.3.2.2 Time Distribution

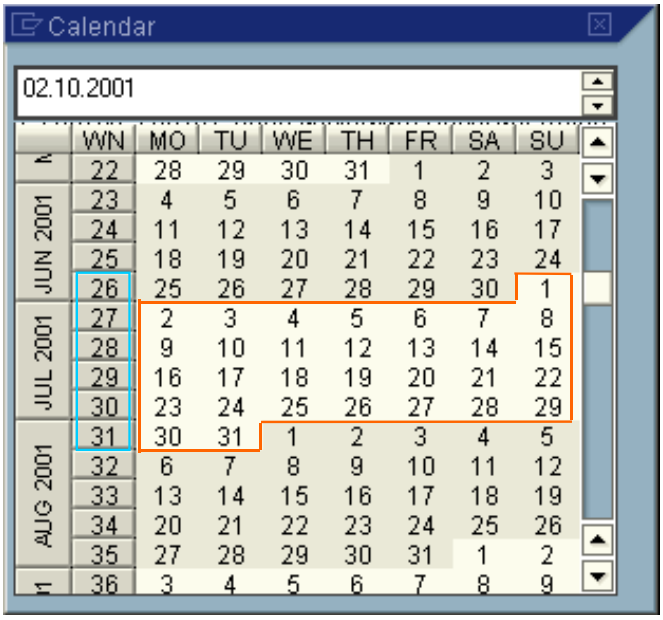
Use

You can update time characteristics with time distribution when you update InfoCubes. All the key figures that can be added are distributed to the corresponding smaller time units.

You can choose time distribution if the InfoSource contains a time characteristic (such as 0CALMONTH) that is more approximate than a time characteristic of the InfoCube (such as 0CALWEEK):

For example, the calendar month 07.2001 would be divided into the weeks 26.2001, 27.2001, 28.2001, 29.2001, 30.2001 and 31.2001. Each key figure that can be added receives 1/31 of the original value for week 200126, 7/31 for each of weeks 27,28,29 and 30, and exactly 2/31 of it for week 31.

The example makes more sense if you compare it to the calendar below:



Functions

You cannot select the time distribution by key figure, it is valid for all key figures.

You can define whether you want the time distribution to be based on the normal calendar or on a particular factory calendar.

Within an InfoCube you may only use one and the same characteristic and one calendar for time distribution for different time characteristics. Otherwise it would not be clear how the different results are to be combined.

For example if 0CALQUARTER is used for distribution on 0CALWEEK and 0CALYEAR for distribution on 0CALMONTH, and 0CALQUARTER contains the value '1.2000' and 0CALYEAR '1999', it would not be clear how the values for 0CALWEEK (01.2000, 02.2000....) should be merged with the values for 0CALMONTH (01.1999, 02.1999...). Only 0CALQUARTER or 0CALMONTH can be chosen.

This function is not implemented for ODS objects.

4.1.7.8.2.3.2.3 Splitting Key Figure Values

Use

You can create several update rules for each key figure in an InfoCube and each data field in an ODS object.

In a company, a sales employee generates a particular volume of sales revenue. In the InfoCube, you want to assign 90% of this sales revenue to the

employee (routine 1) and 10% to the employee's immediate superiors (routine 2).

There are two methods that you can use to split key figure values:

1. Copy Rules

You copy an update rule. You can then assign a different update method to this copy of the update rule. This also gives you the option of updating a key figure with different keys.

In the above example, you create an update rule with routine 1 (90% of the sales revenue assigned to the employee in the InfoCube). Copy the update rule. Maintain routine 2 (10% of the sales revenue assigned to the employee's immediate superiors) in the copy of the rule, and make the necessary changes in the *Characteristic Calculation* tabstrip.

2. Routine with Return Table

Update routines generally only have one return value. However, in the *Key Figure Calculation* tabstrip, you can create a routine by choosing the *Return table* option. The corresponding key figure routine now has a return table instead of a return value. You can now generate as many key figure values as you like from one data record.

In the routine editor, the calculated characteristic values are stored in the ICUBE_VALUES structure. Change these values accordingly (in the above example: *Employee*), fill the field for the relevant key figure (in the above example: *Sales revenue*), and use this to fill the RESULT_TABLE return table.

If you are familiar with ABAP programming, you should use this option, because it gives you a better understanding of how a key figure is updated.

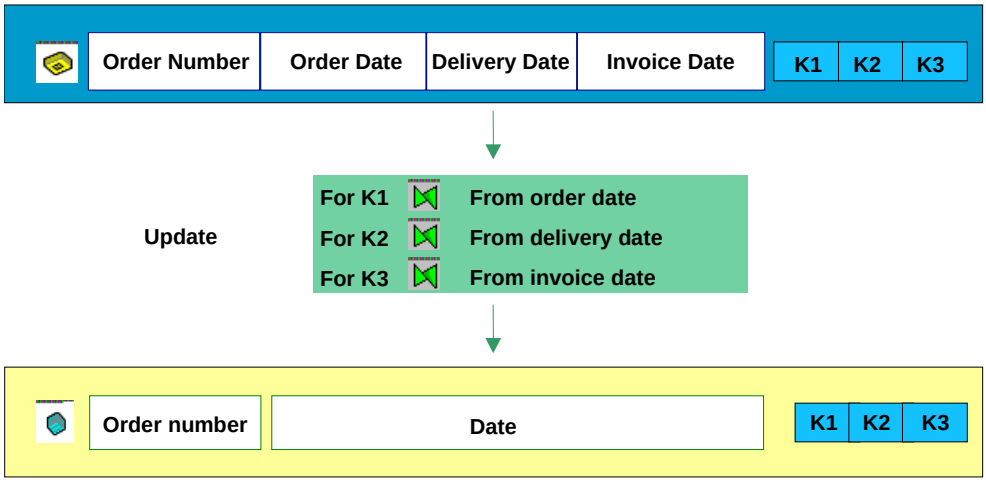
4.1.7.8.2.3.2.4 Creating Different Rules for Different Key Figures for a Characteristic

There are three date characteristics in the InfoSource:

- Order date
- Delivery date
- Invoice date

The InfoProvider only contains a general date characteristic that should be filled from the different date characteristics of the InfoSource, depending on the key figure/data field.

Different rules can be created for the update rule for the date characteristic. Depending on the key figure/data field, the rule updates the order date, the delivery date, or the invoice date in the InfoProvider.



4.1.7.8.2.3.2.5 Currency Translation in the Update

Use

Currency translation in the update allows you to translate data records from the source currency of the InfoSource into a target currency in the data target. This function is only available if key figures with currency fields exist in the source system in various currencies. Currency translation is not available if the currency of the data records to be transferred is the same as the currency of the target key figure in the InfoCube.

Currency translation in the update is normally performed using previously defined translation types. See also Currency Translation Types [External]

It can take place using user-defined subprograms (formulas and routines), if necessary. For ODS objects it is currently not possible to execute the currency translation by using predefined translation types. You have to use routines instead.

Depending on the type of currency, there are two different types of key figures:

1. Key figures with a **fixed currency** (for example, DEM, USD)

   With a fixed currency, the currency is fixed for the amount. The key figure refers specifically to the currency (for example DM, US$), so the currency does not have to be entered again in the data record.

2. Key figures with a **variable currency** (for example, foreign currency).

   A variable currency refers to an InfoObject (for example, `0DOC_CURRCY`).

See also:

Creating InfoObjects: Key Figures [Page 107]

Features

With the update of key figures, there are two basic different cases:

1. For every key figure of an InfoCube (target key figure) there is a corresponding key figure in the InfoSource (source key figure). No currency translation takes place.

2. There is **no** corresponding source key figure in the InfoSource for the target key figure of the InfoCube.

   a. A source key figure of the **same type** can be assigned to the target key figure (for example, sales revenue instead of sales quantity revenue).

      - If the currencies of both of the key figures are the same, no currency translation can take place.
      - If the currencies are different, a translation can take place either using a currency translation type or by simply assigning a currency.

      The following table provides an overview of possible combinations with **different currencies** in the source and target key figures:

Source key figure currency	Target key figure currency	Currency translation (CT)
fixed	variable	No CT
fixed	fixed	CT
variable	fixed	CT
variable	variable	CT or assignment possible

   b. If there is **no** corresponding source key figure of the same type, then you have to fill the key figure for the data target from a **routine**.

      - If the target key figure has a **fixed currency**, no currency translation is carried out. This means that if translation is required, you have to execute it in the routine.
      - If the target key figure has a **variable currency**, you also have to assign a variable source currency to the routine. Using the F4 help you can select a currency from the variable currencies in the communication structure. You have two options:

        – You can select a variable currency and assign it.

        – You select a currency translation type and a currency into which you wish to translate (‘To’ currency).

        The ‘To’ currency is, by default, the target currency if it is included in the communication structure.

Creating a Routine for Currency Translation:

If you want to translate currencies in the update even though the currency translation is not available for one of the above reasons, you can create a routine. Choose *Routine*, set the *Unit Calculation in the Routine* and choose *Create Routine*. In the routine editor you get the additional return parameter UNIT, the value of which is used to determine the target currency.

See also:

Currency Translation [External]

4.1.7.8.2.3.3 Update Routines and Start Routines

Use

You use routines to define complex update rules for a key figure or characteristic.

Routines are ABAP programs that consist of a predefined global data declaration part and an ABAP form routine. The form routine contains all the ABAP programming functions.

You can use the following types of routines in the update:

- The start routine - this routine is run exactly once for a data package at the beginning of the update program
- Routines for updating key figures with and without a return table
- Routines for updating key figures and their unit
- Routines for updating characteristics

Functions

Routines for characteristics and key figures without a return table

The form routine has the following parameters:

- **`MONITOR`**: Available for user-defined monitoring (see also Error Handling in the Update Routine [Page 318])
- **`MONITOR_RECNO`**: The record number is specified for monitoring.
- **`COMM_STRUCTURE`**: The routine for the record that you want to update from the communications structure is stored in the *COMM_STRUCTURE* variable.
- **`RECORD_NO`**: Variable *RECORD_NO* contains, in the form of the communications structure, the number of the transaction data record that is currently to be edited.
- **`RECORD_ALL`**: Variable *RECORD_ALL* contains the number of transaction data records in the form of the communications structure.
- **`SOURCE_SYSTEM`**: Variable *SOURCE_SYSTEM* contains the technical name of the source system.
- **`RESULT`**: Variable *RESULT* must be assigned the result of the computed key figure or computed characteristic.
- **`RETURNCODE`**: You use the *RETURNCODE* variable to control how the data target is written to. The computed *RESULT* value is only written to the data target if *RETURNCODE* is equal to 0. If *RETURNCODE* is not equal to 0, there is an error in the calculation. With key fields, the calculation is terminated, meaning that no data record is determined for this key figure and it is skipped. Data fields are initialized.
- **`abort:`** If *abort* is not equal to 0, the entire update process is canceled.

  In the following example, the loading process is canceled if the value for the InfoObject AMOUNT is less than 0. In this case, the message ‘100’ of type ERROR (E) with the message class ‘ZJH’ appears.

ERROR-type messages also appear in the monitor when the status of the IDoc is displayed.

```
1.  * if abort is not equal zero, the update process will be canceled
2.  if comm_structure-amount < 0.
3.    monitor-msgid = 'ZJH'.
4.    monitor-msgty = 'E'.
5.    monitor-msgno = '100'.
6.    monitor-msgv1 = comm_structure-amount.
7.    append monitor.
8.    abort = 1.
9.    exit.
10. else.
11.   abort = 0.
12. endif.
13. * result value of the routine
14. result = comm_structure-amount.
```

Routines for Key Figures with Unit Calculation

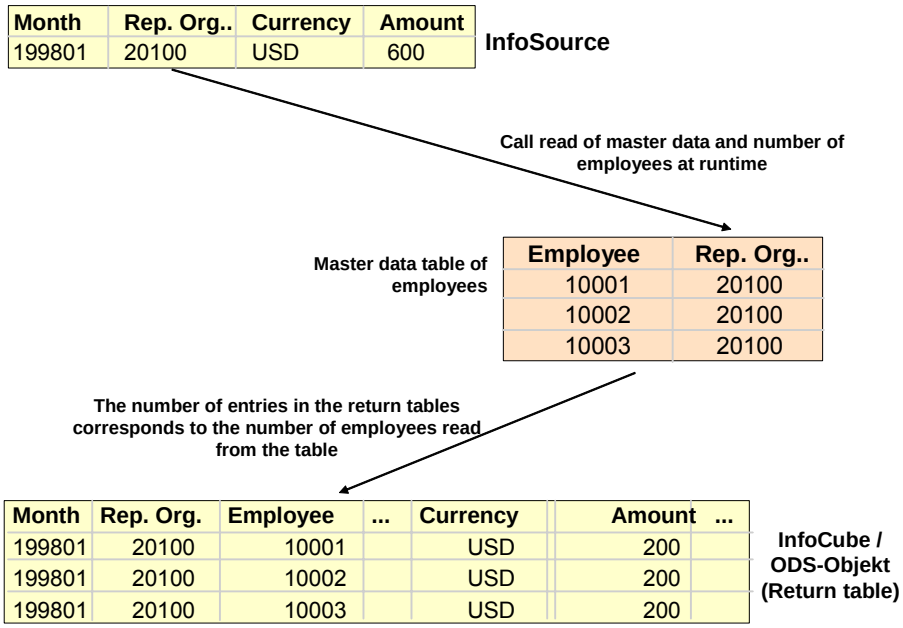
If you set the *Routine with Unit Calculation* flag, the additional UNIT return parameter is also available. There you can store the desired unit for the key figure, for example DEM or ST. See also Currency Translation in the Update [Page 312].

Routines for key figures with return table

For routines with a return table, the RESULT parameter is replaced with the RESULT_TABLE return table. The return table has the same structure as the data target. However, only the key figure or data field, for which the routine was created, is filled, plus its characteristics or key fields. This allows you to distribute aggregated values.

The InfoSource delivers the sales figures for the individual sales organizations. However, the sales need to be displayed for each employee. The following processing steps are required here:

- Read, from the master data table prepared by the InfoSource, the employee IDs that belong to the sales organization.
- Divide the amount of sales by the number of employees that have been read from the master data table.
- Create an entry in your return table for each employee.



The ICUBE_VALUES parameter is also available. The calculated characteristic values are stored here. The easiest way to fill the RESULT_TABLE return table, is to copy the ICUBE_VALUES structure into the RESULT_TABLE table as often as is necessary, and only fill the corresponding key figure and adjust the characteristic values.

Note the following special case: If you use a key figure routine with return table and additional currency translation, the currencies are NOT stored in ICUBE_VALUES structure, because they are only calculated once the routine has been executed.

Start routine

The start routine is run for each data package at the start of the update. The start routine has only these parameters: MONITOR, MONITOR_RECNO, DATA_PACKAGE, RECORD_ALL, SOURCE_SYSTEM and ABORT. DATA_PACKAGE refers to the entire data package. The start routine does not have a return value. It is used to perform preliminary calculations and store these in a global data structure or in a table. You can access this structure or table from other routines.

In the update, you want to credit a certain percentage of the sales turnover to the individual salespersons of a sales office. For example, salesperson A 20% of the turnover of sales office A, salesperson B 30%, salesperson C 50%. The salespersons and their sales offices are listed in a master data table. The uploaded data contains the sales office and sales figures as characteristics. To be able to assign the salespersons to the sales offices, you have to read the data from the master data table on demand. If the master data table is not comprehensive enough, you can generate an internal table in the start routine that contains the master data and that can access the subsequent key figure and characteristic routines.

You also have the option of changing data (for example, modifying or deleting it) by using ABAP code to change the DATA_PACKAGE table.

See also:

Creating Update Routines [Page 317]

4.1.7.8.2.3.3.1 Creating Update Routines

Prerequisites

You are in the update rules maintenance screen for your InfoProvider in the detail view of the key figure/data field/attribute for which you want to create a routine.

Procedure

1. Choose *Routine.*
2. Choose *Create routine.*

   The ABAP editor appears with the following source code:

```
1.  PROGRAM UPDATE_ROUTINE.
2.
3.  *$*$ begin of global - insert your declaration only below this line  *-*
4.  * TABLES:...
5.  * DATA:  ...
6.  *$*$ end of global - insert your declaration only before this line   *-*
7.  *
8.  FORM compute_characteristics
9.    TABLES   monitor structure rsmonitor "user defined monitoring
10.   USING    COMM_STRUCTURE LIKE /BIC/CS<infosource>
11.            RECORD_NO LIKE SY-TABIX
12.            RECORD_ALL LIKE SY-TABIX
13.            SOURCE_SYSTEM LIKE RSUPDSIMULH-LOGSYS
14.   CHANGING RESULT LIKE /BIC/V<infocube>T-<infoobject>
15.            RETURNCODE LIKE SY-SUBRC.
16. *
17. *$*$ begin of routine - insert your code only below this line        *-*
18. * fill the internal table "MONITOR" to make monitor entries
19.
20. * result value of the routine
21.   RESULT =.
22.
23. * if the returncode is not equal zero, the result will not be updated
```

```
24.   RETURNCODE = 0.
25.
26.  * if abort is not equal zero, the update process will be canceled
27.
28.   ABORT = 0.
29.  *$*$ end of routine - insert your code only before this line        *-*
30.  *
31.  ENDFORM.
```

Text in *blue italics* and marked <> is replaced automatically with the relevant object names.

3. In lines 4 and 5, between `*$*$ begin of global...` and `*$*$ end of global...` you can add global declarations (tables or data) that you are able to use in any subsequent routines that you create.

   When you load data in series, this global data remains, provided that the process is retained. For this reason, a REFRESH statement has to be here with internal tables. When you load data in parallel, the global data does not remain, because a separate process is generated for each data packet. See also: Tab Page: Processing [Page 350].

4. Insert your code under `*$*$ begin of routine ...` (line 17) and `*$*$ end of routine ...` (line 29). The *Result* = (21) field contains the result of your update routine.
5. Use the *Check* function to check the syntax of your routine.
6. Save your entries. You end the maintenance session for the routine by leaving the editor.

4.1.7.8.2.3.3.2 Error Handling in the Update Routine

In an update routine, you have the option of transferring error messages and warnings to the monitor.

Note the following:

- When you use the update routine to transfer messages to the monitor, you need to maintain in the scheduler the settings that control how the system behaves if an error occurs. See also Treating Data Records with Errors [Page 373].
- If you set the RETURNCODE = 0 in your routine, the record is posted. If you transfer X-messages, A-messages, or E-messages to the monitor, the record is written to the error request at the same time, because the monitor table contains error messages. With warnings (W-messages) no entry is made in the error request.
- Controlling error behavior with ABORT:

  If ABORT <> 0 the loading process is terminated.

- If, you set the RETURNCODE <> 0 in your routine, there are two possible outcomes, depending on whether you are dealing with a characteristic (key field in an InfoCube) or a key figure (a data field):
  - With key fields, the data record is skipped and the next data record along is processed instead.
  - With key figures, the processing of the key figures is terminated, and the content of the corresponding key figure is initialized and transferred to the target table.

    This process presents no problems so long as you update in an additive way, since this simply involves adding a 0. If, however, you use the *minimum* and *overwrite* update methods, this process becomes problematic. In these cases, you do not use a RETURNCODE, because if you do, any existing entries will be initialized, meaning that they will be overwritten with an empty or blank value.

4.1.7.8.2.3.4 Changing Update Rules

Prerequisites

You have already created an update rule.

Procedure

1. Choose the relevant *InfoProvider* → the relevant *Update rule*→ *right mouse button*→ *Change update rules*.

   The existing update rules are displayed.
2. Select a key figure/data field by double-clicking on it, or by choosing *Detail*.
3. Change the rules for the InfoProvider in your
   - Update type [Page 305]
   - Update method [Page 307]
   - Update routine [Page 317]
4. Use the *Version Comparison* function to compare the modified update rules with the active update rules.
5. Save and activate your entries.

4.1.7.8.2.3.5 Additional Functions in Update Rule Maintenance

Person Responsible

You have the option of specifying a person responsible for a specific update maintenance.

Sorting Rules

You can sort the rules that control the calculation of key figure for routines. It makes sense to do this, for example, if the result of a routine is to be used for additional routines.

The key figures are calculated together with the particular characteristics that you entered in the list sequence.

You can sort rules by selecting one or more rules with the mouse, and then clicking with the left mouse button on a key figure. You can insert the selected rule, or rules, into the rows by choosing *Sort.*

Checking Rules

By using *Check Rules,* you can check whether you have correctly created a rule that belongs to an update rule. If errors are determined during the check procedure, you receive an error message with a marker for the particular maintenance dialog.

Check

By using *Check,* you can activate a check procedure for all rules belonging to an update rule. A status display with traffic light function follows.

4.1.8 Data Transfer Process

Definition

Object that determines how data is transferred between two persistent objects.

Use

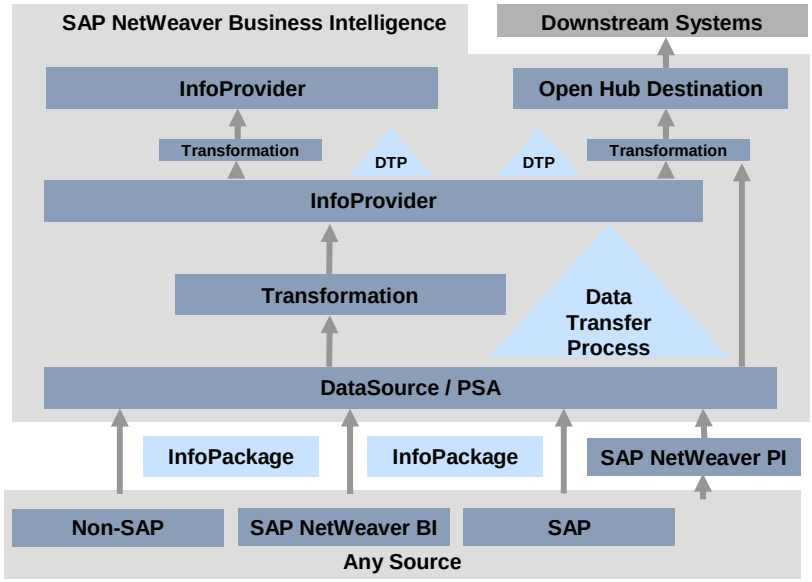
You use the data transfer process (DTP) to transfer data within BI from a persistent object to another object in accordance with certain transformations and filters. In this respect, it replaces the data mart interface and the InfoPackage. As of SAP NetWeaver 2004s, the InfoPackage only loads data to the entry layer of BI (PSA).

The data transfer process makes the transfer processes in the data warehousing layer more transparent. Optimized parallel processing improves the performance of the transfer process (the data transfer process determines the processing mode). You can use the data transfer process to separate delta processes for different targets and you can use filter options between the persistent objects on various levels. For example, you can use filters between a DataStore object and an InfoCube.

Data transfer processes are used for standard data transfer, for real-time data acquisition, and for accessing data directly.

Integration

The following graphic illustrates how the data transfer process is positioned in relation to the objects in the dataflow of BI and the other objects in BI process control:



The InfoPackage controls the transfer of data from the source to the entry layer of BI. The data transfer process controls the distribution of data within BI. The graphic illustrates an example of a data update from the DataSource to an InfoProvider. The data can be updated from an InfoProvider to another InfoProvider using a data transfer process. The data transfer process can also be used to control data distribution from a BI system into any target outside of the BI system. For this purpose, a data transfer process with an open hub destination is used as the target.

Features

You use a process chain to define a data transfer process. Alternatively, you can define a data transfer process for an InfoProvider in an object tree in the Data Warehousing Workbench. We recommend that you use process chains. In this case, the data transfer process is executed when it is triggered by an event in the predecessor process in the process chain. Alternatively, in process chain maintenance, you can execute a data transfer process in the background. A debug mode is also available.

The request is an instance that is generated at the runtime of the data transfer process. The request is processed in the steps that have been defined for the data transfer process (extraction, transformation, filter and so on). The monitor for the data transfer process request shows the header information, request status, and the status and messages for the individual processing steps.

With a data transfer process, you can transfer data either in full extraction mode or in delta mode. In full mode, the entire dataset of the source is transferred to the target; in delta mode, only the data that was posted to the source since the last data transfer is transferred. The data transfer process controls delta handling and therefore allows you to fill several targets with different deltas from one source.

The data transfer process supports you in handling data records with errors. The data transfer process also supports error handling for DataStore objects. When you define the data transfer process, you can determine how the system responds to errors. At runtime, the incorrect data records are sorted and written to an error stack (request-based database table). A special error DTP further updates the data records from the error stack into the target. It is easier to restart failed load processes if the data is written to a temporary storage after each processing step. It also allows you to find records that have errors. In the monitor for the data transfer process request or in the temporary storage for the processing step (if filled), you can

display the data records in the error stack . In data transfer process maintenance, you determine the processing steps after which you want to store data temporarily.

4.1.8.1 Creating Data Transfer Processes

Use

You use the data transfer process (DTP) to transfer data from source objects to target objects within BI. You can also use the data transfer process to access InfoProvider data directly.

Prerequisites

You have used transformations to define the data flow between the source and target object.

Procedure

Creating Data Transfer Processes Using Process Chains

You are in the plan view of the process chain that you want to use for the data transfer process.

The process type *Data Transfer Process* is available in the *Loading Process and Postprocessing* process category.

1. Use drag and drop or double-click to include the process in the process chain.
2. To create a data transfer process as a new process variant, enter a technical name and choose *Create*.

   The dialog box for creating a data transfer process appears.
3. Select the type of data transfer process:

   DTP for Direct Access is the only available option for VirtualProviders. For more information, see Creating Data Transfer Processes for Direct Access [Page 327].

   For DataStore objects, choose *Standard DTP* and *DTP for Real-Time Data Acquisition*. For more information, see Creating Data Transfer Processes for Real-Time Data Acquisition [Page 649].
4. Select the target and source object.

   First select the *object type*.

   When you select the source or target object, input help is available. Input help shows you the selection of objects that already exist in the data flow for the source or target object. If only one object exists in the data flow, this is selected by default.

   An additional *List* pushbutton is available. This allows you to select an object from the complete list of BI objects.
5. Choose *Continue*.
6. The data transfer process maintenance screen appears.

   The header data for the data transfer process shows the description, ID, version, and status of the data transfer process, along with the delta status.
7. On the *Extraction* tab page, define the parameters:

a. Choose extraction mode.

   You can choose full mode or delta mode. You do not need to initialize the delta process explicitly for the delta transfer.

b. If necessary, determine filter criteria for the delta transfer. To do this, choose *Filter.*

   This means that you can use multiple data transfer processes with disjunctive selection conditions to efficiently transfer small sets of data from a source into one or more targets, instead of transferring large volumes of data. You can specify individual selections, multiple selections, intervals, selections based on variables, or routines. Choose *Change Selection* to change the list of InfoObjects that can be selected.

   The icon next to pushbutton *Filter* indicates that predefined selections exist for the data transfer process. The quick info text for this icon displays the selections as a character string.

c. Choose *Semantic Groups* to specify how you want to build the data packages that are read from the source. To do this, you determine key fields. Data records that have the same key are combined in one data package. Currently, only data that has been read from the PSA can be processed further in semantic groups.

   This setting also determines the key fields for the error stack.

   For more information, see Handling Data Records with Errors [Page 328].

   During parallel processing of time-dependent master data, the semantic key of the DTP may **not** contain the field of the data source.

d. Define if the source requests should only be transferred once when updating the delta.

   Setting this indicator ensures that the content of the InfoProvider is an exact representation of the source data.

   A scenario of this type may be required if you always want an InfoProvider to contain the most up-to-date data set for a query, but technical reasons prevent the DataSource on which it is based from delivering a delta (new, changed or deleted data records). For this type of DataSource, the current data set for the required selection can only be transferred using a *full update*.

   In this case, a DataStore object cannot usually be used to determine the missing delta information (overwrite and creation of delta). If this is not logically possible because, for example, data is deleted in the source without delivering reverse records, you can set this indicator and perform a *snapshot scenario*. Only the most up-to-date request for the DataSource is retained in the InfoProvider. Earlier requests for the DataSource are deleted from the (target) InfoProvider before a new one is requested (this is done by a process in a process chain, for example). They are not transferred again during the DTP delta process. When the system determines the delta when a new DTP is generated, these earlier (source) requests are seen as *already fetched*.

e. Specify whether you want to get the data from the source by request.

   Since a DTP bundles all transfer-relevant requests from the source, it generates large requests in some instances. If you do not want to use a single DTP request to transfer the dataset from the source because the dataset is too large, you can set the *Get Data by Request* indicator. This specifies that you want the DTP to read only one request from the source at a time.

You can change this indicator at any time, even if data has already been transferred. If you set this indicator, you can transfer data by request as a one-off activity. If you deselect the indicator, the DTP goes back to transferring all new source requests at once at periodic scheduled intervals.

Note that if you set this indicator, requests may back up in the source during delta update if more requests are posted in the source than are updated using the DTP.

f. Apply any further settings that are relevant for the source object.

8. On the *Update* tab page, define the parameters:

   a. Make the settings for error handling. You define:

      - How you want to update valid records when errors occur.
      - How many errors can occur before the load process terminates.
      - Whether the system gives error status to a load process if records are aggregated, filtered out, or added in the transformation.

      For more information about error handling settings, see Handling Data Records with Errors [Page 328].

   b. Apply any further settings that are relevant for the target object.

9. On the *Execute* tab page, specify the parameters:

   On this tab page, the process flow of the program for the data transfer process is displayed in a tree structure.

   a. Specify the status that you want the system to adopt for the request if warnings are to be displayed in the log.

   b. Specify how you want the system to define the overall status of the request.

   c. Normally the system automatically defines the processing mode for the background processing of the respective data transfer process.

      If you want to execute a delta without transferring data, analogously to the simulation of the delta initialization with the InfoPackage, select *No data transfer; delta status in source: fetched* as *processing mode*. This processing mode is provided when the data transfer process extracts in delta mode. A request that is started in this way marks the data that is found in the source as fetched, without actually transferring it to the target.
      If delta requests have already been transferred for this data transfer process, you can still choose this mode.

10. Check the data transfer process, save and activate it.

Additional Functions

Choose *Goto* → *Overview of DTP* to display information about the source and target objects, the transformations, and the last changes to the data transfer process.

Choose *Goto* → *Batch Manager Settings* to make settings for parallel processing with the data transfer process.

Choose *Goto* → *WHM Statistics Settings* to make settings for runtime measurements with data transfer processes.

Creating Data Transfer Processes from the Object Tree in the Data Warehousing Workbench

The starting point when creating a data transfer process is the target into which you want to transfer data. In the Data Warehousing Workbench, an object tree is displayed and you have highlighted the target object.

1. In the context menu, choose *Create Data Transfer Process*.

   The dialog box for creating a data transfer process appears.

2. Proceed as described in steps 3 to 10 in the procedure for creating a data transfer process using a process chain.

   In step 4 you specify the source object only.

Result

When you return to process chain maintenance, the data transfer process of the plan view is displayed and you can include it in your process chain. When you activate and schedule the chain, the system executes the data transfer process as soon as it is triggered by an event in the predecessor process in the chain.

If you do not include the data transfer process in a process chain, you can execute it in the background.

If you want to execute the data transfer process in debugging mode, choose processing mode *Serially in the Dialog Process (for Debugging)*. In this case, you can define breakpoints in the overview for the process flow of the program. The request is processed synchronously in a dialog process and the update of the data is simulated. If you select *expert mode*, you can also define selections for the simulation and activate or deactivate intermediate storage in addition to setting breakpoints. For more information see Simulating and Debugging DTP Requests [Page 335].

4.1.8.2 Creating Data Transfer Processes for Real-Time Data Acquisition

Use

You use the data transfer process (DTP) for real-time data acquisition to transfer data to the DataStore object from the PSA. In the DataStore object, the data is available for use in reporting.

Prerequisites

You have used transformations to define the data flow between the DataSource and the DataStore object.

The selections for the data transfer process do not overlap with selections in other data transfer processes.

Procedure

The starting point when creating a data transfer process is the DataStore object into which you want to transfer data. In the Data Warehousing Workbench, an object tree is displayed and you select the DataStore object.

1. In the context menu, choose *Create Data Transfer Process*.

   The dialog box for creating a data transfer process appears.

2. Select *DTP for Real-Time Data Acquisition* as the *DTP Type*.

3. As the source object, select the DataSource from which you want to transfer data to the DataStore object.

   The input help for the source object shows the selection of DataSources that already exist in the data flow for the DataStore object. An additional *List* pushbutton is available. This allows you to select a DataSource from the complete list of BI DataSources.

4. Choose *Continue*.

   The data transfer process maintenance screen appears.

   The header data for the data transfer process shows the description, ID, version, and status of the data transfer process, along with the delta status.

5. On the *Extraction* tab page, specify the parameters:

   a. *Delta* is chosen as the extraction mode for real-time data acquisition.

   b. If necessary, determine filter criteria for the delta transfer. To do this, choose *Filter.*

      This means that you can use multiple data transfer processes with disjunctive selection conditions to efficiently transfer small sets of data from a source into one or more targets, instead of transferring large volumes of data. You can specify individual selections, multiple selections, intervals, selections based on variables, or routines. To change the list of InfoObjects that can be selected, choose *Change Selection*.

      The icon next to pushbutton *Filter* indicates that predefined selections exist for the data transfer process. The tool tip for this icon displays the selections as a character string.

   c. Semantic grouping is not used for real-time data acquisition. The data is read from the source in packages.

6. On the *Update* tab page, specify the parameters:

   Make the settings for error handling. You determine:

   - How you want to update valid records when errors occur.
   - How many errors can occur before the load process terminates.

     Note that this setting only has an impact while repairing a DTP request (repair) and during the conversion of the DTP to standard DTP (for example, to correct an error during extraction).

     For real-time data acquisition, specify in the InfoPackage the maximum number of failed attempts that is permitted when the daemon is accessing data before the data transfer terminates with an error.

For more information about error handling settings, see Handling Data Records with Errors [Page 328].

7. On the *Execute* tab page, specify the parameters:

   On this tab page, the process flow of the program for the data transfer process is displayed in a tree structure.

   a. Specify the status that you want the system to adopt for the request if warnings are to be displayed in the log.

   b. Specify how you want the system to determine the overall status of the request.

8. Check, save, and activate the data transfer process.

Result

To access the monitor for real-time data acquisition, choose *Real-Time Monitor*. In the context menu of a DataSource, you can assign the data transfer process to a daemon. The daemon uses the data transfer process to process data.

4.1.8.3 Creating Data Transfer Processes for Direct Access

Use

You use a data transfer process for direct access to access the data in an InfoProvider directly.

Prerequisites

You have used transformations to define the data flow between the source and target object.

Procedure

The starting point when creating a data transfer process is the target into which you want to transfer data. In the Data Warehousing Workbench, an object tree is displayed and you have highlighted the target object, a VirtualProvider.

1. In the context menu, choose *Create Data Transfer Process*.

   The dialog box for creating a data transfer process appears.

   DTP for Direct Access is displayed as the type of the data transfer process.

2. Select the type of source object.

   Supported object types are DataSources, InfoCubes, DataStore objects and InfoObjects (texts and attributes, if they are released as InfoProviders).

3. Select the object from which you want to transfer data into the target.

   When you select the source object, input help is available. Input help shows you the selection of objects that already exist in the data flow for target object. If only one object exists in the data flow, this is selected by default.

An additional *List* pushbutton is available. This allows you to select a source object from the complete list of objects that exist for this object type.

4. Choose *Continue*.

   The data transfer process maintenance screen appears.

   The header data for the data transfer process shows the description, ID, version, and status of the data transfer process, along with the delta status.

   On the *Extraction* tab page, the system displays information about the adapter, the format of the data and additional source-specific settings.

   On the *Update* tab page, the system displays information about the target.

   On the *Execute* tab page, the system displays the processing mode for direct access and the process flow of the program for the data transfer process.

   You do not need to make any settings in the data transfer process.

5. Check the data transfer process, save and activate it.

Choose *Goto* → *Overview of DTP* to display information about the source and target objects, the transformations, and the last changes to the data transfer process.

Result

You can use the data transfer process to access data directly.

4.1.8.4 Handling Data Records with Errors

Use

On the *Update* tab page in the data transfer process (DTP), the *error handling* settings allow you to control how the system responds if errors occur in the data records when data is transferred from a DTP source to a DTP target.

These settings were previously made in the InfoPackage. When using data transfer processes, InfoPackages only write to the PSA. Therefore, error handling settings are no longer made in the InfoPackage but in the data transfer process.

Features

For a data transfer process (DTP), you can specify how you want the system to respond when data records contain errors. If you activate error handling, the records with errors are written to a request-based database table (PSA table). This is the error stack. You can use a special data transfer process, the error DTP, to update the records to the target. You create an error DTP for an active data transfer process on the *Update* tab page. You run it directly in the background or include it in a process chain so that you can schedule it regularly in the context of your process chain. The error DTP uses the full update mode to extract data from the error stack (in this case, the source of the DTP) and transfer it to the target that you have already defined in the data transfer process. Temporary storage is available after each processing step of the DTP request. This allows you to determine the processing step in which the error occurred.

The following table provides an overview of where checks for incorrect data records can be run:

Where Is Check Run?	Examples of Incorrect Data Records
In the transformation	Field contains invalid characters or lowercase characters Error during conversion Error during currency translation A routine returns a return code <> 0 Characteristic value is not found for master data Error reading master data Customer-specific formula results in error
When data is updated to the master data table or text table	Invalid characters in keys or navigation attributes If no SID exists for the value of the navigation attribute If no language field is defined for texts Invalid "from" and "to" dates Duplicate data records with relation to keys Overlapping and invalid time intervals
When data is updated to the InfoCube	If no SID exists for the characteristic value
When checking referential integrity of an InfoObject against master data tables or DataStore objects	If no SID exists for the characteristic value

Activities

1. On the *Extraction* tab page under *Semantic Groups*, define the key fields for the error stack. The key should be as detailed as possible. A maximum of 16 key fields is permitted. The fewer the number of key fields defined, the more records are updated to the error stack.

   This setting is particularly relevant if you are transferring data to DataStore objects with data fields that are overwritten. If errors occur, all subsequent data records with the same key are written to the error stack along with the incorrect data record; they are not updated to the target. This guarantees the serialization of the data records, and consistent data processing.

   If fields in the DataStore object are overwritten, the system automatically marks all source fields that are assigned uniquely to a key field of the target as a proposal for the key fields of the error stack. Fields that are not uniquely assigned to key fields of the target can also be marked as key fields for the error stack. Source fields that are assigned to a data field of the target cannot be marked as key fields for the error stack.

   The system does not provide a proposal if the DataStore object contains only fields that are updated cumulatively.

   See also: Error Stack [Page 330] and Examples for Using the Error Stack [Page 331]

2. On the *Update* tab page, specify how you want the system to respond to data records with errors:

   a. No update, no reporting (default)

If errors occur, the system terminates the update of the entire data package. The request is not released for reporting. However, the system continues to check the records.

b. Updating valid records, no reporting (request red)

This option allows you to update valid data. This data is only released for reporting after the administrator checks the incorrect records that are not updated and manually releases the request (by a QM action, that is, setting the *Overall* status on the *Status* tab page in the monitor).

c. Updating valid records, reporting possible

Valid records can be reported immediately. Automatic follow-up actions, such as adjusting the aggregates, are also carried out.

3. Specify the maximum number of incorrect data records that are allowed before the system terminates the transfer process.

If you do not make an entry here, handling for incorrect data records is not activated and the update is terminated when the first error occurs.

4. Under *No Aggregation*, select how you want the system to respond if the number of data records received differs from the number of data records updated.

A difference between the number of records received and the number of updated records can occur if the records are sorted, aggregated, or added in the transformation rules or during the update.

If you set this indicator, the request is interpreted as incorrect if the number of received records differs from the number of updated records.

If the number of selected records differs from the number of records received, this is interpreted as an error regardless of whether or not the indicator is set.

5. Make the settings for the temporary storage by choosing *Goto* → *Settings for DTP Temporary Storage*. In these settings, you specify the processing steps after which you want the system to temporarily store the DTP request (such as extraction, filtering, removing new records with the same key, and transformation). You also specify when you want to clear the temporary storage. Under *Level of Detail*, you specify how you want to track the transformation.

6. Once the data transfer process has been activated, create an error DTP on the *Update* tab page and include it in a process chain. If errors occur, start it manually to update the corrected data to the target.

4.1.8.4.1 Error Stack

Definition

A request-based table (PSA table) into which erroneous data records from a data transfer process are written. The error stack is based on the data source, that is, records from the source are written to the error stack.

Use

At runtime, erroneous data records are written to an error stack if the error handling for the data transfer process is activated. You use the error stack to update the data to the target destination once the error is resolved.

Integration

In the monitor for the data transfer process, you can navigate to the PSA maintenance by choosing *Error Stack* in the toolbar, and display and edit erroneous records in the error stack.

With an error DTP, you can update the data records to the target manually or by means of a process chain. Once the data records have been successfully updated, they are deleted from the error stack. If there are any erroneous data records, they are written to the error stack again in a new error DTP request.

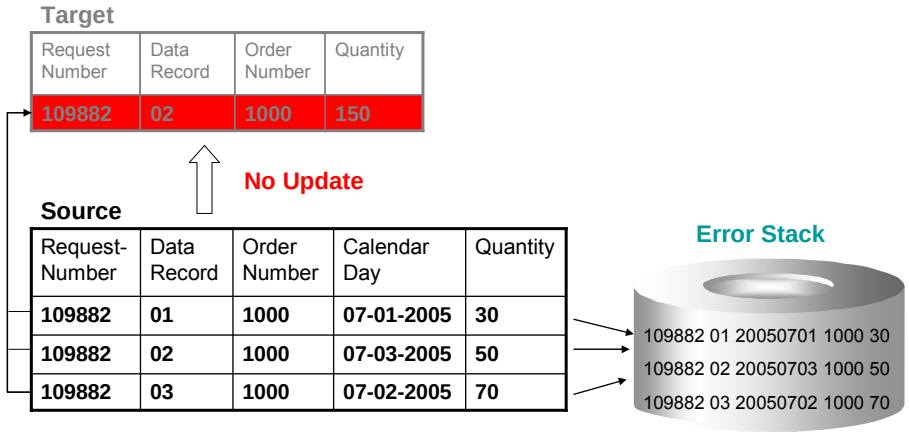
When a DTP request is deleted, the corresponding data records are also deleted from the error stack.

4.1.8.4.2 Examples for Using the Error Stack

Consistent Error Handling for Aggregation

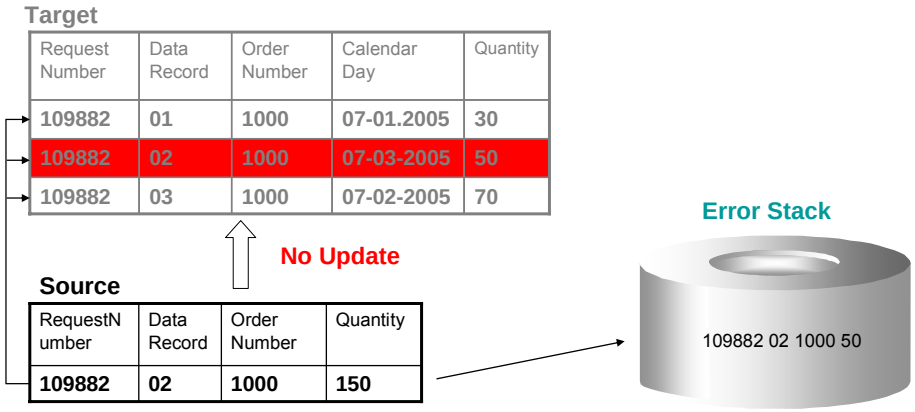
Number of Records in Source is Greater than Number of Records in Target

During the transformation, the data records for request *109882* are aggregated to one data record. If, for example, there is no SID for the characteristic value order number *1000*, the record is interpreted as erroneous. It is not updated to the target. Those data records that form the aggregated data record are written to the error stack.



Number of Records in Source is Less than Number of Records in Target

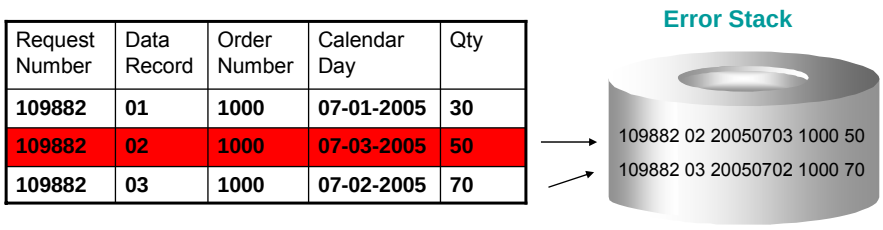
During the transformation, the data record for request *109882* is duplicated to multiple data records. If, for example, there is no SID for the characteristic value calendar day *07-03-2005*, the record is interpreted as erroneous. The duplicated records are not updated to the target. The data record that formed the duplicate records is written to the error stack. In the error stack, the record is listed as containing an error every time it duplicates data records with errors.



Consistent Error Handling with Respect to Order in Which Data Records are Written to Error Stack

Update to DataStore Object: 1 Request

The *Order Number* field is the key for the error stack. During the transformation, data record *02* of request *109882* is marked as containing errors. In addition to the erroneous data record, all subsequent data records for the request that contain the same key are written to the error stack. In this example, this is data record *03*. This ensures that when error records are updated with the error DTP, the records are serialized correctly and newer data is not inadvertently overwritten by older data.

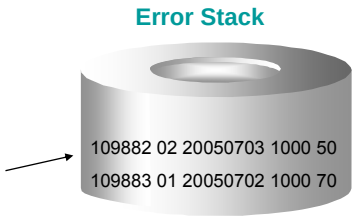


Updating to DataStore Object: Multiple Requests – Error in First Request

The *Order Number* field is the key for the error stack. During the transformation, data record *02* of request *109882* is marked as containing errors. In addition to the erroneous data record, all subsequent data records, including the following requests, that have the same key are written to the error stack. In this example, data record *01* for request *109883* is written to the error stack in addition to data record *02* for request *109882*.

Request Number	Data Record	Order Number	Calendar Day	Quantity
109882	**01**	**1000**	**07-01-2005**	**30**
109882	**02**	**1000**	**07-03-2005**	**50**

Request Number	Data Record	Order Number	Calendar Day	Quantity
109883	**01**	**1000**	**07-02-2005**	**70**

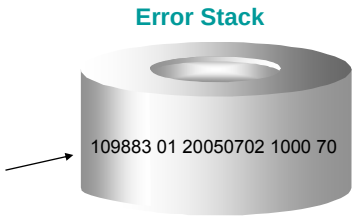


Updating to DataStore Object: Multiple Requests – Error in Subsequent Request

The *Order Number* field is the key for the error stack. During the transformation, data record *01* of request *109883* is identified as containing errors. It is written to the error stack. Any data records from the previous request that have the same key were updated successfully to the target.

Request Number	Data Record	Order Number	Calendar Day	Quantity
109882	**01**	**1000**	**07-01-2005**	**30**
109882	**02**	**1000**	**07-03-2005**	**50**

Request Number	Data Record	Order Number	Calendar Day	Quantity
109883	**01**	**1000**	**07-02-2005**	**70**



4.1.8.4.3 emporary Storage

Definition

Table that contains the data records processed with the request in a specific DTP processing step. The table contains correct records as well as erroneous records with a status display.

Use

The temporary storage is used at runtime to allow processes to be restarted without a significant impact on performance when errors occur. It also allows you to find records with errors. Unlike on the source level, this is only possible if, under *Level of Detail*, you have set *Trace of Erroneous Records* or *Trace of Transformation by Records*.

Integration

You can display the temporary storage for each of the processing steps using the monitor for the data transfer process request by choosing the pushbutton in the *Data* column. The records with errors can be identified by the symbol.

In data transfer process maintenance, you can specify when the temporary storage is to be deleted. This can be done either after the request has been updated successfully to the target, when the request is deleted, or after a specific interval has passed since the request was processed.

4.1.8.5 Monitor

Use

A request is generated to execute your data transfer process. The monitor allows you to keep track of the request in the various processing steps.

Integration

The request status and request messages from the data transfer processes (DTPs), and the InfoPackage load processes are integrated into the transfer monitor, which you can call in the *Administration* functional area of the Data Warehousing Workbench by choosing *Monitors*.

You can also call the monitor using data transfer process maintenance, using the process monitor on a DTP in the log view of the process chain maintenance, or using the *Call Monitor* function for a target object in an object tree in the Data Warehousing Workbench.

Features

The monitor for a data transfer process request displays detailed information about the processing of the request, in addition to the header information.

At the first level of the detail tree, the system displays the status of the requests at various points during the process, for example, at generation, when processing the steps during the DTP program, and when setting the technical and complete status of the request. The status display for the processing is then divided further according to the steps of the data transfer process and displays the status and messages for each of the processing steps and data packages in the processing steps.

At the processing step level, you can also branch into the temporary storage display for this particular processing step (if you have defined in DTP maintenance that the temporary storage is to be filled after this processing step). You can branch to the PSA maintenance for the error stack if data has been written to the error stack and has not been updated yet.

From the monitor, you can branch to the where-used list for the request. This allows you to display independent target requests and delete requests consistently.

You can also branch to the administration for the source and the target (DataSource or InfoProvider), as well as to the job and process overviews.

With *Debugging* you can Simulate and Debug DTP Requests [Page 335].

4.1.8.5.1 Where-Used List for Requests

Use

The where-used list supports you in finding dependent requests in further targets. The where-used list therefore allows you to delete across several persistence layers.

Integration

You can call the where-used list for DTP requests in InfoCubes and DataStore objects if the request has already been loaded into other data targets. You can also call the where-used list for requests in the PSA tables of the DataSource (R3TR RSDS).

Features

The where-used list displays the source request and all related target requests in a hierarchy. You can delete requests directly from the where-used list display.

4.1.8.5.2 imulating and Debugging DTP Requests

Use

Simulating the data update and debugging a data transfer process request (DTP request) help you to analyze an incorrect DTP request.

In this way you can also simulate a transformation prior to the actual data transfer if you would like to check whether it provides the desired results.

Procedure

Simple Simulation and Debugging

1. If you want to execute a standard DTP in debug mode, choose *Serially in the Dialog Process (for Debugging)* as the processing mode on the *Execute* tab page in the DTP maintenance.
2. In the overview for program execution, define breakpoints for debugging by choosing **Change Breakpoints**.
3. Choose **Simulate**.

   The simulation request is started. The request is processed synchronously in a dialog process. If you defined breakpoints, you go to the ABAP Debugger.

   The result of the simulation is displayed in the DTP monitor.

Simulation and Debugging in Expert Mode

In addition to setting breakpoints, expert mode enables you to specify selections for simulating the request and for defining the substep after which you want to write to temporary store during the simulation.

1. You can perform the simulation of a request in expert mode from the DTP maintenance or the DTP monitor.

a. In the DTP maintenance, select the processing mode *Serially in the Dialog Process (for Debugging)* on the **Execute** tab page, set the indicator **Expert Mode,** and choose **Simulate**.

b. In the DTP monitor, choose **Debugging**.

The *Debug Request* screen appears.

2. Make the settings on the *Filter* tab page.

When you call the simulation from the DTP monitor, the filter settings that are displayed as initial are copied from the request displayed in the monitor. When you call it from the DTP maintenance, the filter settings displayed as initial are copied from the next “real” request of the DTP.

- You can change the filter in order to restrict the data for the analysis. In addition to the selection with the fields of the source, you can also restrict the request ID (REQUID), data package ID (DATAPKID), and data record number (RECORD) with the technical fields. You should restrict the data in order to prevent the simulation from running for a long time, for example, if you loaded a large amount of data. Selections are also recommended if the fields refer to certain data records in the source or to a certain selection of data in the source.
- With **Save as Variant** you can store the selections in the database for later use.
- With **Load Variant** you can call a selection variant stored for this request.
- With **Technical Field Name / Description** you can switch between displaying the technical field name and the description.
- With **Change Selection** you can change the list of fields displayed in the selection screen.

3. Make your settings on the **Temporary Storage and Breakpoints** tab page.

You can define if and when to write data to temporary storage during processing and set breakpoints at different times.

4. Choose **Execute**.

The simulation request is started. The request is processed synchronously in a dialog process. If you defined breakpoints, you go to the ABAP Debugger.

The result of the simulation is displayed in the monitor and can be analyzed.

4.1.9 Scheduler

Use

You use the scheduler for the load process to define the parameters for the data transfer to a particular DataSource. You determine what sort of data (flow data, master data, texts, or hierarchies), from which DataSource, and from which source system are requested and written to the BI entry layer, the Persistent Staging Area.

Activities

You create an InfoPackage for the data request in the scheduler for a DataSource. You determine the parameters for scheduling the data request for a DataSource in the InfoPackage

We recommend including InfoPackages in the process chains that automate the processes in the Data Warehouse.

4.1.9.1 Creating InfoPackages

Use

If your data flow is modeled with transformations, if you are using a new or emulated DataSource, the InfoPackage loads into the PSA only. Data is distributed within the BI system with the data transfer process.

Only the parameters that are not already defined in the DataSource are defined on the tabs of the InfoPackage maintenance. Parameters that refer to the target of the data update that follows the PSA are defined in the data transfer process.

Procedure

You are in an object tree in the Data Warehousing Workbench.

1. Mark the required DataSource and choose *Creating InfoPackages* in the context menu.
2. You get to the scheduler of the BI system.
3. On the *Data Selection* tab, define the selection criteria for the data transfer. If there are hierarchies, define which hierarchy you want to load on the *Hierarchy Selection* tab.
4. The *Extraction* tab page shows you the settings made in the DataSource for the adapter and data format.
5. On the *Processing* tab page, define if the consistency check should be performed for characteristics. Under *Parallelism*, define how the parallel processing of the processes should be performed. Since the data is only updated in the PSA, no further definitions are required to update the data.
6. On the *Update* tab page, define the update mode if required. The possible selections are displayed depending on the DataSource.
7. If required, define if the data should also be updated if there is no master data for the data or if no data update should be performed as soon as no master data exists for a characteristic.
8. If necessary, define the time selection for time-dependent data.
9. Save the InfoPackage.

Result

You can now use the InfoPackage in a process chain. It is available there as a process variant for process type *Execute InfoPackage*. Alternatively you can schedule the InfoPackage in the scheduler. We recommend its use in process chains.

See also:

Tab Page: Data Selection [Page 346]

Tab Page: Hierarchy Selection [Page 349]

Tab Page: Processing [Page 350]

Tab Page: Update [Page 358]

Scheduling InfoPackages [Page 365]

4.1.9.2 Additional Scheduler Functions

- Testing delete selection at start.

  If you want to Automatically Delete or Cancel Requests for the InfoCube when loading [Page 357], via *Scheduler → Test Delete Selections at Start,* you can determine whether the requests are to be deleted or canceled with or without the system displaying a warning.

- Displaying initialization for a DataSource

  Under *Scheduler → Initialization Selection for Source System,* the system displays all initializations for the DataSource.

  You can find additional information from Tabpage: Updating [Page 358] in the **Initialization of Delta Process** section.

- Control Parameters Maintenance for Data Transfer [Page 370]

  With the settings made via *Scheduler → DataS. Default Values for Data Transfer*, you can override the settings you made in Customizing for a particular DataSource and a particular update mode.

- Indicating request as repair full request

  You can use a request that was selected as a repair request via *Scheduler → Repair Full Request* to carry out a full update in any data target. This also applies to data requests that already contain data from an initialization run or deltas for this DataSource/source system combination, and that have overlapping selection criteria.

  You can find additional information from Tabpage: Updating [Page 358] in the **Full Update** section.

- Setting time-out time

  You can use *Scheduler → Time-out Time* to set the maximum wait time after which the system determines the request as being in error. If the system has not received any (additional) data record before the maximum wait time has elapsed, the request assumes the status to be incorrect. You can use this function to override system-wide settings for a particular InfoPackage. The default setting for each InfoPackage is the same as the system-wide setting you made under *Monitor → Settings* (or in BW Customizing).

- Setting up warning handling

  Using *Scheduler → Warning Handling*, you can determine how the system reacts to warnings, should they occur, that is, whether the request gets the status green, yellow or red in such a case. You can use this function to override system-wide settings for a

particular InfoPackage. The default setting for each InfoPackage is the same as the system-wide setting you made under *Monitor → Settings* (or in BW Customizing).

We recommend you **do not** assign the status yellow to the request when a warning arises, since this prevents subsequent processing steps from starting. The request would then be denoted as incorrect after the time-out time had elapsed anyway.

.

4.1.9.2.1 Maintaining Data Transfer Control Parameters for DataSources

Use

In the BI Scheduler you have the option of determining control parameters for data transfer for individual DataSources. This function is available with *Scheduler → Default DataSource Settings for Data Transfer*. You can update transaction data in the PSA in this way, for example in larger data packages. Smaller packets guarantee faster processing if you want to update master data in the dialog.

You can define the following parameters for DataSources from SAP source systems:

Control Parameter	Update Mode
Size of data package	For all possible update modes
Number of parallel processes for the data transfer	For all possible update modes
Frequency with which the status IDocs are sent	For all possible update modes
Maximum number of data packages per request If there is an initial value, the system performs the data transfer according to the settings in the source system. A value 0 overwrites the setting in the source system. In this case there is no limit.	For delta and repeat requests

You can define the following parameters for file DataSources:

Control Parameter	Update Mode
Size of data package	For all possible update modes
Frequency with which the info IDocs are sent	For all possible update modes

Integration

DataSources from SAP Source Systems

The settings you made in the Scheduler override the settings made in extractor-customizing for SAP source systems for a particular DataSource and a particular update method.

In general you maintain the control parameters for data transfer from an SAP source system in the Customizing of the extractors. You go to Customizing of the extractors from the context

menu of the corresponding source system in the source system tree of the Data Warehousing Workbench. Choose *Business Information Warehouse → General Settings → Maintaining Control Parameters for Data Transfer* to display or change the settings for the data transfer on source system level.

File DataSources

Use those settings that you made in the Scheduler to control the setting that you made under *Business Intelligence → Links To Other Systems → Maintain Control Parameters For Data Transfer* for a particular DataSource.

4.1.9.3 Handling Duplicate Data Records

DataSources for texts or attributes can transfer data records with the same key into BI in one request. Whether the DataSource transfers multiple data records with the same key in one request is a property of the DataSource. There may be cases in which you want to transfer multiple data records with the same key (referred to as duplicate data records below) to BI more than once within a request; this is not always an error. BI provides functions to handle duplicate data records so that you can accommodate this.

Dataflow with Transformation

In a dataflow that is modeled using a transformation, you can work with duplicate data records for time-dependent and time-independent attributes and texts.

If you are updating attributes or texts from a DataSource to an InfoObject using a DTP, you can specify how you want the system to handle data records that have the same record key. In DTP maintenance on the *Update* tab page, you set the *Handle Duplicate Record Keys* indicator.

This indicator is not set by default.

If you set the indicator, duplicate data records (multiple records with identical key values) are handled as follows:

- Time-independent data:

  If data records have the same key, the last data record in the data package is interpreted as being valid and is updated to the target.

- Time-dependent data

  If data records have the same key, the system calculates new time intervals for the data record values. The system calculates new time intervals on the basis of the intersecting time intervals and the sequence of the data records.

  Data record 1 is valid from 01.01.2006 to 31.12.2006

  Data record 2 has the same key but is valid from 01.07.2006 to 31.12.2007

  The system corrects the time interval for data record 1 to 01.01.2006 to 30.06.2006. As of 01.07.2006, the next data record in the data package (data record 2) is valid.

  If you set the indicator for time-dependent data, note the following:

You cannot include the data source field that contains the DATETO information in the semantic key of the DTP. This may cause duplicate data records to be sorted incorrectly and time intervals to be incorrectly calculated.

The semantic key specifies the structure of the data packages that are read from the source.

Example

You have two data records with the same key within one data package.

In the following graphic, DATETO is not an element of the key:

	Key	DATETO	DATEFROM	Value
Rec. 1	A	31.12.2006	01.01.2001	xyz
Rec. 2	A	31.12.2001	01.01.2000	abc

In the data package, the data records are in sequence DS2, DS1. In this case, the time interval for data record 1 is corrected:

Data record 1 is valid from 1.1.2002 to 31.12.2006.

Data record 2 is valid from 1.1.2000 to 31.12.2001.

In the following graphic, DATETO is an element of the key:

	Key		DATEFROM	Value
		DATETO		
Rec. 1	A	31.12.2006	01.01.2001	xyz
Rec.2	A	31.12.2001	01.01.2000	abc

If DATETO is an element of the key, the records are sorted by DATETO. In this case, the data record with the earliest date is put before the data record with the most recent date. In the data package, the data records are in sequence DS2, DS1. In this case, the time interval for data record 2 is corrected:

Data record 2 is valid from 1.1.2000 to 31.12.2000.

Data record 1 is valid from 1.1.2001 to 31.12.2006.

If you do not set this indicator, data records that have the same key are written to the error stack of the DTP.

Dataflow with 3.x Objects

In a dataflow that is modeled using 3.x objects, you can only work with duplicate data records for time-independent attributes and texts.

If the DataSource transfers potentially duplicate data records, this information is passed to the scheduler when you create new InfoPackages. On the *Processing* tab page, the *DataSource Transfers Duplicate Data Records* indicator is set. The *Ignore Duplicate Data Records* indicator is also set. When multiple data records that have the same key are transferred, the last data record in the request is updated to BI. Any other data records in the request with the same key are ignored.

To maintain consistency, ignoring duplicate data records is only possible if the data is updated serially. A serial update is when data is first updated into the PSA then, after it has been successfully written to the PSA, it is updated into the master data or text tables of the InfoObject.

If a DataSource transfers potentially duplicate data records or if you manually set the *Ignore Duplicate Data Records* indicator, the *PSA Only* update type is automatically selected in the scheduler.

If you deselect the *Ignore Duplicate Data Records* indicator, any duplicate data records cause errors. The error message is displayed in the extraction monitor.

Since the characteristic key is used to identify duplicate data records, make sure that all the key fields of a characteristic are filled by the DataSource. If this is not the case, for example because key fields are hidden in the extract structure, the system notifies you of this when you activate the transfer rules.

Note

You can specify how duplicate data records within a request are handled, independently of whether the setting that allows DataSources to deliver potentially duplicate data records has been made. This is useful if the setting is made for the DataSource but the system knows from other sources that duplicate data records are transferred (when loading flat files, for example).

4.1.9.4 cheduler (3.x)

Definition

The scheduler is the connection between the source systems and the SAP Business Information Warehouse. You use the scheduler to determine when and from which InfoSource, DataSource, and source system, data (transaction data, master data, texts or hierarchies) is requested and updated.

The principle behind the scheduler relates to the functions of SAP background jobs. The data request can be scheduled either straight away or it can be scheduled with a background job and started automatically at a later point in time.

In the *Modeling* functional area in the Administrator Workbench, you access the data request from the scheduler by choosing *InfoSource Tree → Your Application Component → Your InfoSources → Your Source System → Context Menu (right mouse button) → Create InfoPackage.*

Use

Data is requested for a combination of application components, an InfoSource [Page 270], a DataSource, and a source system.

When requesting data, create an InfoPackage [Page 343] in the InfoSource tree in the Administrator Workbench. In the InfoPackage, you set the conditions for the scheduling of the data request Therefore an InfoPackage is always assigned to just one InfoSource, one DataSource to the data type that goes with it (transaction data, master data, texts, or only ever one hierarchy), and to one source system.

When you create an InfoPackage, this is saved as administrative information with the present variant settings [Page 369].

Integration

The scheduler helps to load and update the data into the BW system. You can check the BW loading and processing processes in the Administrator Workbench Monitor [External]. The data is available for analysis after the BW processing process.

4.1.9.4.1 Maintaining InfoPackages

Use

With the help of an InfoPackage, you determine the selection conditions in BW for the data request from a source system. In this section, you can select the contents of the InfoPackages so that it can be scheduled.

If you do not enter any specific selections, then all of the available data for the DataSource will be requested from the source system. You only have to make a selection when dealing with hierarchies.

If you have previously selected a delta process then you only get the affected delta for your selection. There is a maximum of one active delta selection for a combination of InfoSource, DataSource and source system.

If you change already scheduled InfoPackages, your new selection is used in later background jobs, for example, periodic runs. You have to create a new InfoPackage if you want to continue to use the existing InfoPackage.

If you access an external system with the data request, you must make the data available in special formats. You can find additional information about this in the sections Maintaining InfoSources (Flat Files) [Page 286] and Maintaining InfoSources (External Systems) [Page 300]).

InfoPackages have a connection to the BW transport system [External] and document management [External].

Procedure

1. Highlight a source system in the InfoSource tree in the Administrator Workbench.
2. Choose *Context menu (right mouse click)* → *Create InfoPackage* and enter any alphanumeric or numeric name for the InfoPackage.

   Dialog box that follows will show you which source system you want to load data and to which InfoSource. You will see the DataSources that are assigned to the InfoSource to which data is to be loaded.
3. Select a DataSource for which you want to load data into BW.

If the InfoPackage is scheduled, data is loaded into BW from the DataSource selected here.

Now go through the procedure for the relevant system:

Procedure for SAP Systems [Page 344]

Procedure for Flat Files [Page 344]

Procedure for External Systems with BAPI Functions [Page 345]

4.1.9.4.1.1 Procedure for SAP Systems

InfoPackage with Transaction Data

1. Maintain the tab page: Data Selection [Page 346]
2. Maintain the tab page: Processing [Page 350]
3. Maintain the tab page: Data Targets [Page 356].
4. Maintain the tab page: Updating [Page 358]
5. Schedule the InfoPackage [Page 365].

InfoPackage with Texts or Master Data

1. Maintain the tab page: Data Selection [Page 346]
2. Maintain the tab page: Processing [Page 350]
3. Also maintain the tab page Data Targets [Page 356] for master data and texts, when updating flexibly.
4. Maintain the tab page: Updating [Page 358]
5. Schedule the InfoPackage [Page 365].

InfoPackage with Hierarchies

1. Maintain the tab page: Hierarchy Selection [Page 349].
2. Maintain the tab page: Processing [Page 350]
3. Maintain the tab page: Updating [Page 358]
4. Schedule the InfoPackage [Page 365].

See also:

Data Extraction from SAP Source Systems [Page 523]

4.1.9.4.1.2 Procedure for Flat Files

InfoPackage with Transaction Data

1. Maintain the tabpage: External Data [Page 362].
2. Maintain the tabpage: Data Selection [Page 346]
3. Maintain the tabpage:Processing [Page 350]
4. Maintain the tabpage: Data Targets [Page 356].
5. Maintain the tabpage: Updating [Page 358]
6. Scheduling InfoPackages [Page 365].

InfoPackage with Texts or Master Data

1. Maintain the tabpage: External Data [Page 362].
2. Maintain the tabpage: Data Selection [Page 346]
3. Maintain the tabpage:Processing [Page 350]
4. Also maintain the tab strip Data Targets [Page 356] for master data and texts, when updating flexibly.
5. Maintain the tabpage: Updating [Page 358]

6. Schedule the InfoPackage [Page 365].

InfoPackage with Hierarchies

1. Maintain the tabpage: External Data [Page 362].
2. Maintain the tabpage: Hierarchy Selection [Page 349].
3. Maintain the tabpage:Processing [Page 350]
4. Maintain the tabpage: Updating [Page 358]
5. Schedule the InfoPackage [Page 365].

See also:

Maintaining InfoSources (flat files) [Page 286]

4.1.9.4.1.3 Procedure for External Systems with BAPI Functions

InfoPackage with Transaction Data

1. Maintain the tabpage: Data Selection [Page 346]
2. Make the corresponding BAPI-specific selections.
3. Maintain the tabpage:Processing [Page 350]
4. Maintain the tabpage: Data Targets [Page 356].
5. Maintain the tabpage: Updating [Page 358]
6. Scheduling InfoPackages [Page 365].

InfoPackage with Texts or Master Data

1. Maintain the tabpage: Data Selection [Page 346]
2. Make the corresponding BAPI-specific selections.
3. Maintain the tabpage:Processing [Page 350]
4. Also maintain the tabpage Data Targets [Page 356] for master data and texts, with flexible updates.
5. Maintain the tabpage: Updating [Page 358]
6. Scheduling InfoPackages [Page 365].

InfoPackage with Hierarchies

1. Maintain the tabpage: Hierarchy Selection [Page 349].
2. Make the corresponding BAPI-specific selections.
3. Maintain the tabpage:Processing [Page 350]
4. Maintain the tabpage: Updating [Page 358]
5. Scheduling InfoPackages [Page 365].

See also:

Maintaining InfoSources (External System) [Page 300]

4.1.9.4.1.4 Tab Page: Data Selection

Use

Determine the selection values for the InfoPackage in the *Data Selection* tab page.

You can determine selection boundaries not only for InfoObjects mapped in the transfer structure, but also for unmapped fields in the source system. The fields are displayed in the *Technical Name in Source System Displayed* column in the selection table. To tell the difference between InfoObjects and fields when the selections are displayed, the fields are shown in brackets (for example, with the initialization selection for source systems in the Scheduler).

You can either enter single values or intervals directly in the *from value* and *to value* fields or set the selections as variable. You can find additional information under Variable Change of Selections with Background Processing [Page 348].

When using selection criteria, InfoObject or field data corresponding to the selection limit is requested from the source system. With InfoObjects or fields for which no variable selections were determined, all available data is requested from the source system.

You can specify several selections for an InfoObject/field in which you insert duplicate rows. Additional rows for entries in this InfoObject/field are then inserted into the selection table. When selecting individual value- or interval selections and not confirming them, you can select an additional variable selection type in the same row of the InfoObject/field. The system automatically adds an additional row in the selection table for this selection. Thus, your selections for an InfoObject or field are displayed in a clear way.

Selection Options for DataSource from SAP Source Systems

When using a DataSource from an SAP source system, the DataSource determines which selection options are possible. All DataSources permit

- Single value selections (`l_t_range-sign = 'I'` and `l_t_range-option = 'EQ'`)
- Intervals (`l_t_range-sign = 'I'` and `l_t_range-option = 'BT'`).

Note that additional selection options are only possible if they are supported by the respective DataSource.

You can specify model selections (`l_t_range-option = 'CP'`), such as **`ABC*`**, in the field *From Value* as a single value. Note the following:

- As a model, **only** use the character *.
- Use the character * only **once** in the field *From Value*.
- Use the character * only **at the end of the value**.

If you execute a selection with * in the *From Value* field, and the *To Value* field is not filled, check the scheduler to see if the DataSource allows the selection option `CP`. If this is the case, the `l_t_range-option` is changed from `EQ` to `CP`. `If this is` **`not`** `the case, the scheduler returns an info message and passes the data request with the selection option EQ to the source system.` If you fill the field *To Value*, the scheduler **always** passes the data request with the selection option `BT` to the source system and does not execute any check. The request is not cancelled in these cases.

You can also specify selection options for the data request by means of a routine or a variable. If a selection that is not supported by the DataSource is transferred from a routine or variable to the scheduler, the entire request is cancelled.

You can find additional information about selection options with routines and variables under Variable Changes for Selections in the Background Processing [Page 348].

You can display the selection options supported by your DataSource in the source system in the data browser (SE16). Specify the **ROOSFIELD** table and choose *Table* → *Table Contents*. The *SELOPTS* field includes the value that the combination of selection options supported by the DataSource characterizes. For more information, see the F1 help for *SELOPTS*.

Selection Limits for Data from File- and External- Systems.

It is also possible to determine selection criteria for data from files and external systems. Only records that meet the selection criteria are loaded.

To determine field selections in the Scheduler, firstly highlight them in the transfer structure maintenance in the column *Selection* and save and activate the transfer structure.

By determining the selection criteria, you can carry out the functions *Automatic Deletion* and Checking Data Consistency of the Requests in the InfoCube [External] for data from files and external systems.

Selection Limits for Updating from an ODS Object

When wanting to update data from an ODS object into additional data targets using full-update, you can set selection criteria for this schedule.

You can find additional information under Update ODS Object [External].

Conversion Exits

When entering the selection, the conversion exits displayed in the selection table are executed in the OLTP and the results are temporarily saved in the BW. If a selection has already been converted, it is in the BW; if it is not yet in the BW, conversion takes place in the OLTP.

If conversion cannot take place in the OLTP, because the system cannot be accessed or the OLTP conversion exit is not running properly, conversion takes place in the BW. If this is not possible either, the *Conversion Exit* field is ignored, meaning that the selections must be entered in an internal format. In this case, entering in external format will lead to errors in the load process.

If you have set the *Use Conversion Exit* option, the system adds two spaces to the length of each input field that has a conversion exit (except ALPHA exit). You can then use the external format (which is usually longer than the internal) in these fields.

If you want to deactivate the conversion exit in general, remove the *Use Conversion Exit* selection. Here, enter the selections in an internal format too. The system then takes the two additional spaces, that would have been needed for the external format, away from the length of the input field..

Archive Selections

If you reload data from an archive file [Page 766], select your archive file by using the pushbutton *Archive Selection*. For the time being, you can only load archive data within the BW.

Notes

- If a DataSource field is assigned to the InfoObject 0LANGU, the field does not appear as a selection field in the Scheduler, despite being available for selection. In this case, the source system implicitly requests all languages from the language vector active in BW.
- If time-dependent data is loaded, the time interval is determined in the scheduler in the update tab page [Page 358]. The DATETO and DATEFROM fields, as well as the fields that are assigned to the 0DATETO and 0DATEFROM InfoObjects, are therefore not available as selection fields on the *Data Selection* tab page.
- In the selection criteria for the scheduler, only upper case letters are permitted. If you enter a value in lower case letters and confirm this entry, the value is converted into upper case letters. Also, you can only make selections according to lower case letters if you make the selection with the help of a routine

4.1.9.4.1.4.1 Variable Selection Change in Background Processing

Use

If you want to load data into BI from the source system periodically and change the contents of the selection field each time you load, you can define the selections to be variable. You can implement a variable change of selections with each load by using ABAP routines or (OLAP) variables. The variable selections are not replaced by concrete values until a data request is made.

You select the variable type in the column *Type (Variable Change of Selections for Background Processing)*.

Features

Selections using an ABAP routine (variable type 6)

You can create an ABAP program that restricts the requested data of an InfoObject or field. To do this, select type 6 and confirm your selection. A screen appears on which you enter a name for the ABAP routine. After entering the name, the editor appears and can enter your own coding. The ABAP routine has access to all selection fields and is the last to be processed at runtime.

You must explicitly make the following definitions for single value selections and intervals in the routine: For the field `l_t_range-option` = "EQ" or "BT" and for `l_t_range-sign` = 'I'. Note that there is no check whether the field contents are meaningful.

Selections using a variable (variable type 7)

(OLAP) variables are used as placeholders for values of InfoObjects. They are replaced with concrete values during a data request. To use a Variable [External] for restricting the

selection, select type 7 and confirm your selection. A screen appears on which you can select the variables.

The system does not check for data type and length when selecting variables. If errors occur, they are not found until the runtime of the data request. However, you can display the selections for a particular variable by choosing *Check*. The system displays the restrictions that would apply when data is requested in a table. If the variable cannot be used to define the selection conditions, the system raises an error and terminates the action.

The way that selections are filled at runtime depends on the processing type for the variable [External]. For time selections, you cannot use any variables that use a replacement path or require manual input.

4.1.9.4.1.5 Tab Page: Hierarchy Selection

Use

Use this tab page to select the hierarchy that you want to load and make hierarchy-specific settings for the load process.

Functions

The hierarchies for the DataSource of your InfoPackage are in the selection table.

There are three options to store a loaded hierarchy in BW.

- **Full Update**

  The hierarchy is stored under the selected or determined technical name in the BW. A hierarchy with the same technical name in BW is overwritten if necessary.

- **Storing the Hierarchy as a Subtree of a Target Hierarchy**

  The loaded hierarchy is added to a tree that already exists in BW, the **target hierarchy,** as a subtree. The target hierarchy node, to which the root nodes of the **subtree hierarchy** correspond, are called **interface nodes**. It is the node that refers to the same InfoObject in the target and subtree hierarchies. It also has the same technical name.

  There are two options to store subtree hierarchies in BW:

- **Subtree Insert**

  The hierarchy is inserted in an existing hierarchy as a subtree, without nodes being deleted from the target hierarchy.

  If a sub-tree is inserted for a second time, there are two sets of each sub-tree hierarchy node amongst the interface nodes. The results in termination of the load process.

- **Subtree Update**

  The hierarchy is inserted in an existing hierarchy as a subtree, and the old subtree is replaced by the new one if required.

If a subtree update is executed again, all target hierarchy nodes that are found amongst the interface nodes are deleted, before the new subtree is inserted amongst the interface nodes.

You can find further information about inserting and updating subtrees in the Loading Subtree Hierarchies. [External] section.

Activities

1. You can select whether a hierarchy is to be loaded or whether it is to be automatically activated or flagged for activation after being successfully loaded.

   If the selected hierarchy is used in an aggregate, it cannot be activated directly after being loaded. It is flagged for activation and activated when a hierarchy or attribute is next changed.

   An inactive hierarchy cannot be used in queries.

   When using the InfoPackage in a process chain, the indicator for automatic activation or flagged activation is hidden. This is because the setting is represented by the corresponding process type *Save Hierarchy* in process chain maintenance.

2. To avoid overwriting an existing BW query when you load a new hierarchy which has the same name but which comes from a different source system, indicate *Rename Hierarchy After Loading* and enter a new technical name. The hierarchy is stored under this new technical name and the original technical name from the source system is not used.

   Loading an SAP cost center hierarchy with the same name from different source systems.

3. When you want to load a hierarchy from an external system from a file by using BAPI , you firstly need to create a hierarchy header. This is because the file or the external system only delivers the structure of the hierarchy. You cannot select the hierarchy for the load process until the hierarchy header has been created.

   See also:

   Uploading Hierarchies from Flat Files [Page 291]

4. When you create an InfoPackage for a hierarchy from the OLTP, the system displays hierarchies made available by the DataSource in the OLTP for selection. The system saves these entries and makes them available in the selection table again when you want to select hierarchies in this InfoPackage once more. To display the latest OLTP hierarchy entries for the DataSource, refresh the table using the *Available Hierarchies In OLTP* pushbutton.

4.1.9.4.1.6 Tab Page: Processing

Use

You can use this tab page to determine the destination of the updated data.

Prerequisites

If you want to update the data using the PSA, you have set the transfer method to *PSA* in the transfer rules maintenance.

If you want to transfer the data using IDocs, you have set the transfer method to *IDoc* in the transfer rules maintenance.

We recommend you use the PSA transfer method.

Functions

PSA Transfer Method

Different options are available for data update when using the PSA transfer method. Upon selection, you need to weigh data security against performance for the loading process.

Processing options for the PSA transfer method

	Processing option	Description	Further Information
	PSA and Data Targets/Info Objects in Parallel (By Package)	A process is started to write the data from this data packet into the PSA for each data package. If the data is successfully updated in the PSA, a second parallel process is started. In this process, the transfer rules are used for the package data records, data is adopted by the communication structure, and it is finally written to the data targets. Posting of the data occurs in **parallel by packet**. This method is used to update data into the PSA and the data targets with a high level of performance. The BW system receives the data from the source system, writes it to the PSA, and starts the update immediately and in parallel into the corresponding data target.	The maximum number of processes, which is set in the source system in Customizing Extractors [Page 535], does not restrict the number of processes in BW. Therefore, many dialog processes in the BW system could be necessary for the loading process. Make sure that enough dialog processes are available in the BW system. If the data package contains incorrect data records, you have several options allowing you to continue working with the records in the request. You can specify how the system reacts to incorrect data records You can find additional information about this in the section Treating Data Records with Errors [Page 373]. You also have the option of Correcting Data in the PSA [Page 51] and updating it from there. Note the following when using transfer and update routines: If you choose this processing option and then request processing takes place in parallel during loading, the global data is deleted because a new process is used for every data package in further processing.
	PSA and Then into Data Target/Info Object (by Package)	A process that writes the package to the PSA table is started for each data package. When the data has been successfully updated to the	If the data package contains incorrect data records, you have several options allowing you to continue working with the records in the request. You can find additional information about this in the section Handling Data Records with Errors [Page 373].

		PSA, the same process then writes the data to the data targets. Posting of the data occurs in **serial by packet**. Compared with the first processing option, you have better control over the whole data flow with a serial update of data in packages, because the BW system carries it out using only one process for each data package. Only a certain number of processes are necessary for each data request in the BW system. This number is defined in the settings made in the maintenance of the control parameters in customizing for extractors.	You also have the option of Correcting Data in the PSA [Page 51] and updating it from there. Note the following when using transfer and update routines: If you choose this processing option and then request processing takes place in parallel during loading, the global data is deleted because a new process is used for every data package in further processing.
Only PSA		Using this method, data is written to the PSA and is not updated any further. You have the advantage of having data stored safely in BW and having the PSA, which is ideal as a persistent incoming data store for mass data as well. The setting for the maximum number	When using the InfoPackage in a process chain, this setting is hidden in the scheduler. This is because the setting is represented by its own process type in process chain maintenance and is maintained there. Treating Duplicate Data Records [Page 371] (only possible with the processing type *Only PSA*): The system indicates when master data or text DataSources transfer potential duplicate data records for a key into the BW system. The *Ignore Duplicate Data Records*

		of processes in the source system can also have a positive impact on the number of processes in BW. To further update the data automatically in the corresponding data target, wait until all the data packages have arrived and have been successfully updated in the PSA, and select *Update Subsequently in Data Targets* from the *Processing* tab page when you schedule the InfoPackage in the Scheduler. A process that writes the package to the PSA table is started for each data package. If you then trigger further processing and the data is updated to the data targets, a process is started for the request that writes the data packages to the data targets one after the other. Posting of the data occurs in **serial by request**. You can find additional information about the options for uploading data from the PSA into the data targets	indicator is also set by default in this case.. In BW, the last data record of a request is updated for a particular key by default when data records are transferred more than once. The remaining data records are ignored for this key in the same request. If the *Ignore Duplicate Data Records* indicator is not set, duplicate data records will cause an error. The error message is displayed in the monitor. Note the following when using transfer and update routines: If you choose this processing options and request processing takes place serially during loading, the global data is kept as long as the process with which the data is processed is in existence.

		under Type of Data Update [Page 47] in the section **Only PSA → Further Processing from PSA**.	
	Only Data Target/Info Object	We only recommend this method when loading from flat files or from data sources that are always available. This is because the data is not always saved persistently into an incoming data store. This saves hard disk space, although it leads to a loss of the following: data security, an option to rebuild processes, and simulation options.	

IDoc Transfer Method

The following options are available for data update when using the IDoc transfer method:

- ***ALE inbound processing and data target/InfoObject***

  In this method, data is loaded in BW ALE inbound processing, and updated further into the data targets directly.

- ***Only ALE inbound processing***

  In this method, data is loaded in BW ALE inbound processing, but not updated further. The request appears in the monitor as incorrect. You can manually update the data further from here into the data targets.

When you send too many data packages that are too small, too many monitor logs are written for them. This has a negative effect on the monitor run time.

We recommend a data package size of 10,000 to 50,000 data records. With mass data (requests with a size of two to three million records), the size of the data package can be from 50,000 to 100,000 records. However, the required memory depends on not only the data package size setting, but also the width of the transfer structure, the required memory of the affected extractor, and, for large data packages, the number of data records in the package.

See also:

Documentation for Customizing for extractors in the source system, especially the section *Maintaining Control Parameters for Data Transfer*, under *General*

Settings. You reach Customizing for extractors by using the source system tree in the Administrator Workbench – Modeling. Choose *Customizing for Extractors* for your source system by using the context menu (right mouse click) or choose the transaction SBIW directly in your source system.

Consistency Check for Characteristic Values

When you set the *Consistency Check for Characteristic Values* indicator, the system only checks the loading of master data and transaction data and the activation of hierarchies after reading the data from a PSA or IDoc. The InfoObject definition is checked.

The characteristic values are checked for

- The use of small characters,
- The use of special characters,
- The plausibility of date fields,
- The plausibility of time fields,
- The use of character values in the fields of the NUMC data type, and
- Correct observance of the ALPHA conversion routine

.

You should always select this indicator with new data as well as non-SAP data during the test phase. The indicator for the productive area is deactivated if the data was of good quality (several data imports without incorrect values). This is because using this function worsens load performance.

4.1.9.4.1.7 Tab Page: Data Targets

Use this tab page to determine to which InfoCubes or ODS objects, when updating flexibly, to which InfoObject master data- or text table, the data for a data request is updated.

Note for Loading into an InfoCube

You cannot delete indexes, or create statistics or indexes while data is being loaded into an InfoCube. We recommend you serialize these processes using process chains.

You can also use in this screen to get to the update rule maintenance, if the update rules have not been activated yet.

By using *Maintain Data Target Contents* , you can get to the Data Target Administration [External]

The following functions are also available for InfoCubes only:

1. Automatic Deletion of Similar or Same Requests from the InfoCube [Page 357]

   Use the pushbutton in the column *Automatic Deletion of Similar/Same Requests from the InfoCube*to determine the selection criteria to delete data from the InfoCube. After successfully loading the new request, deletion or cancellation is carried out automatically according to the selection criteria. You can only cancel a request if no

exception aggregation has been defined in the InfoCube or in the update rules and all InfoObjects are updated with summation.

2. Completely delete data target content

   When you select the indicator to completely delete the data target content for an InfoCube, the system deletes before loading the data.

   When using the InfoPackage in a process chain [Page 375] , this setting is hidden in the scheduler. This is because the setting is represented by its own process type in process chain maintenance and is maintained there.

   See also:

   Deleting the InfoCube and InfoCube Content [External]

4.1.9.4.1.7.1 Automatic Deletion of Requests from the InfoCube when Loading

Use

To ensure that the data that is going to be loaded with an InfoPackage is available only once in an InfoCube, you can delete from the InfoCube any existing requests that have partially overlapping or similar selection conditions and relevant deletion selections.

After loading new data into the InfoCube, the InfoCube contents are checked using the selection conditions for an InfoSource/DataSource/source system combination, to see if overlapping data has been loaded into the InfoCube. The data that has been loaded into the InfoCube is not analyzed rather the selection conditions for the InfoObjects that you determined in the Scheduler are compared.

With this function the dataset in BW is reduced to the size necessary for analysis, and redundancies are avoided.

Prerequisites

The requests that are deleted or cancelled in the InfoCube must still be available in the PSA.

Functions

You select the variable deletion conditions when you schedule the InfoPackage that is going to load the new data. On the *Data Targets* tab page in the *Automatic Deletion of Similar/Same Requests from InfoCube* column, you click on the pushbutton to go to the *Deletion of Requests from the InfoCube After Update* dialog box.

Here you determine under which conditions existing requests are deleted from the InfoCube:

- You can include update modes, InfoSources, DataSources, and source systems for requests.

  Requests are deleted when the appropriate deletion conditions are met if existing requests and new ones were loaded from the same DataSource.

- You can determine whether the selections for the new request need to be the same as or more comprehensive than the selections of the request that is to be deleted, or

whether the selections for the new request need to completely or partially overlap the selections of the request that is to be deleted.

- You can specify deletion restrictions according to date. This means that, when the deletion conditions are met, only those requests that were loaded within a specific time period are actually deleted.

  If you choose *Data for the Request from the Last Seven Days*, only those requests are deleted that have been loaded in the last seven days.

- You can specify situations in which a request is not to be deleted, even if the appropriate deletion conditions are in place.

  If, under *Exceptions*, you choose the *Today is …the Last Day of a Month*, existing requests are not deleted, if the new request is loaded on the last day of a month.

- Another option for determining deletion conditions is creating a user routine for the InfoPackage and the InfoCube. In this routine you can exclude requests from a deletion process or include additional requests for deletion.

The deletion and cancellation processes are carried out automatically only after the new request has been loaded successfully.

If the request is not compressed, it is deleted. A request is cancelled only if, due to a compression, deletion is no longer an option (because the request number is no longer transparent.) You can only cancel a request if no exception aggregation has been defined in the InfoCube or in the update rules and all InfoObjects are updated with summation.

Under the path *Scheduler → Test Deletion Selections at Start*, you can specify whether or not you want the system to display a warning in the Scheduler when requests are about to be cancelled or deleted.

Process Variant: *Deleting Overlapping Requests from an InfoCube*

When you use the corresponding process in the process chain maintenance screens to specify the conditions for automatic deletion, these conditions are applied to all the InfoCubes for the selected InfoPackages.

In the variants maintenance screen for the *Delete overlapping requests from InfoCube* process, choose the *Execute InfoPackage* object type and select the InfoPackages that you want to use. The InfoPackages you specify here need to be included in the process chain, in which the deletion/cancellation is going to take place.

If you specify in the Scheduler the conditions for automatic deletion, these conditions are applied to the InfoCube that you also selected in the Scheduler.

4.1.9.4.1.8 Tab Page: Updating

Use

On the *Update Parameters* tab page, specify the update mode, the type of update in the data targets, error handling when loading and updating the data further, and time selection for time-dependent data.

Update Mode

The following update functions are available in the scheduler:

Full Update

A full update requests all data that corresponds to the selection criteria you determined in the scheduler.

In a certain period of time specified in the scheduler, 100,000 data records are accumulated. With a full update, all 100,000 data records are requested.

Using the *Scheduler* menu, you can indicate a request with full-update mode as *Repair Full Request*. This request can be updated into every data target, even if the data target already contains data from an initialization run or delta for this DataSource/source system combination, and has overlapping selection criteria.

Note: Using the full repair request to reload the data into the DataStore object after selectively deleting data from the DataStore object can result in inconsistencies in the data target. Updating this request can lead to duplicate data records in the data record when the repair request selections do not agree with the selections from selectively deleting from the DataStore object.

For information about selectively deleting from a DataStore object, see DataStore Object Content [External].

Delta Update

A delta update only requests data which has appeared since the last delta.

Before you can request a delta update, you must first initialize the delta process.

A delta update is only possible for loading from SAP source systems.

If a delta fails (status 'red' in the monitor) or the overall status of the delta request has been set to red manually, the next data request will be performed in repeat mode. A repeat request extracts the incorrect or incompletely loaded data from the failed delta request, along with any new data since the original request. A repeat can only be requested in the dialog. If data from the failed delta request has already been updated to the data targets, delete the data from the data targets affected. If you cannot delete the data from the data targets, duplicate data records may be produced when the repeat request is performed.

Initializing the Delta Process

To request deltas, you need to have initialized the delta process.

For selection criteria that do not overlap, several initialization criteria are possible for initializing the delta process when scheduling InfoPackages for data from an SAP system. This gives you the option of loading the relevant data for the delta process, step by step into the Business Information Warehouse. For example, you can load the data into BI for cost center 1000 in the first step, and the data for cost center 2000 in the second step.

A delta requested after several initializations, contains the sum of all the successful initial selections as a selection condition. This selection condition can then no longer be changed for the delta. In this example, data is loaded into BI for cost centers 1000 and 2000 with a delta.

The selections for initialization are made in the scheduler on the *Select Data* tab page.

You can display all initializations for a DataSource in the scheduler by choosing *Scheduler → Initialization Selection for Source System*. You can see whether an initialization selection has been executed and what the status of the initialization request is. From this dialog box, you can also transfer initialization selections into the scheduler, and delete selected initializations again. Choosing the pushbutton *Initialization Request Status* brings you to the monitor, where you can check the data request.

Deleting initializations that have been run and registered in delta administration from the delta administration

If you delete initializations from the delta administration, the delta administration is reset in BI as well as in the source system. You can only request deltas again if you have run a new initialization for this DataSource/source system combination.

Note:

If you delete initializations from the delta administration, no data is deleted in BI.

Note for DataStore objects:

If the delta process for your DataSource requires serialization, after deleting the initialization and performing a new initialization, you can no longer update this init request into the previously supplied DataStore objects as with all subsequently requested delta requests. This is because the old inits and deltas still exist there.

The system recognizes that there are now new inits and deltas that are not yet updated in the DataStore object and requires that they be updated. However, the selections of the new inits overlap with the selections of the old inits that still exist in the DataStore object, so that the new inits cannot be updated.

If the delta process for the DataSource requires serialization, the system does **not** activate the newly loaded inits or deltas in the DataStore object.

In this case, after init administration has been deleted, you also have to delete the data that was loaded with these inits and their subsequent deltas into the supplied DataStore objects from the DataStore objects, before you can activate new inits and deltas.

For test purposes, you can initialize the delta process without transferring data. In the *Update Parameters* tab page, select the corresponding indicator. When the initialization simulation runs, BW and the source system create the help entries in all required tables, but do not load data. Afterwards, the monitor displays a loaded record. The init simulation request appears in all data targets into which data was updated, but only contains the help entries.

Delta initialization can only be simulated for DataSources from SAP source systems if the DataSource supports this. The corresponding indicator only appears in the *Update* tab page when this is the case.

Generic DataSources, for which you have set a generic delta, also support the simulation of delta initialization.

The option of simulating delta initialization is generally offered for loading processes from other source systems (for example, file or non-SAP systems).

Early Delta Initialization

With early delta initialization, you have the option of writing the data into the delta queue or into the delta tables for the application during the initialization request in the source system.

This means that you are able to execute the initialization of the delta process (the init request), without having to stop the updating of data in the source system. You can only execute an early delta initialization if the DataSource extractor called in the source system with this data request supports this.

Extractors that support early delta initialization were delivered with Plug-Ins as of Plug-In (-A) 2002.1.

You cannot run an initialization simulation together with an early delta initialization.

Build Initial Non-Cumulative

This update method is only available when you want to load data into a data target that contains a non-cumulative key figure.

Choose this update mode when loading non-cumulative data into BI for the first time. For more information, see Transferring Non-Cumulative Data into the BI System [Page 430].

Error Handling

Error handling [Page 373] options allow you to determine system behavior with the following processing steps:

- Transfer rules
- Update rules
- Master data, text, and hierarchy updates
- InfoCube updates

Referential integrity [Page 279] of an InfoObject against master data tables or DataStore objects on the communication structure level.

Time Selection For Time-Dependent Data

Selections for the data that is to be requested with an InfoPackage is specified in the scheduler on the *Data Selection* tab page. The time selections for time-dependent data are specified on the *Update* tab page. The DATETO and DATEFROM fields, as well as the fields that are assigned to the 0DATETO and 0DATEFROM InfoObjects, are therefore not available as selection fields on the *Data Selection* tab page.

Time-dependent DataSources that use generic data extraction:

The time dependency of a DataSource that uses generic data extraction is identified by containing selectable fields with the names DATETO and DATEFROM. Note the following: If, in the table from which you want to extract data, time-dependency is displayed in fields with names other than DATETO / DATEFROM (for example, DATBI / DATAB), you have to create a database view that maps these fields to DATETO / DATEFROM. If the DataSource was already created, regenerate it. Otherwise, changes in the database view will not be transferred into the generated extraction structure.

Time-dependent DataSources for Business Content:

In DataSources that are delivered in Business Content, time-dependency can also be displayed with other field names. In these cases, the respective fields in

the BI system need to be assigned to the InfoObjects 0DATETO and 0DATEFROM. In doing so you ensure that, in terms of the selection, they are treated in exactly the same way as fields with the field names DATETO and DATEFROM are treated during generic extraction.

Various options are available for selecting time-dependent data that you want to load into BI:

- Fixed time interval
- Creating a routine

  You use this routine to specify complex time selections for data to be requested, for example, if you only want to load the data for specific times that cannot be specified with an interval or a variable.
- Using variables (see Variables [External])

  Variables are placeholders for values and are not replaced by concrete values until a data request is made. For example, you use 0CMONTH variables to load the data for the current calendar month.

The way that selections are filled at runtime depends on the processing type for the variable [External]. For the time selections, you cannot use any variables that use a replacement path or require manual entry.

The system does not check for data type and length when selecting variables. Should errors occur, they are not recognized until the runtime of the data request. However, you can display the selections for a particular variable by choosing *Check*. The system displays the selections that will be determined when data is requested in a table. If the variable cannot be used to determine the selection conditions, an error is produced and the display is terminated.

We recommend that you create variables (in the BEx Query Designer [External]) for time selection only if you want to use the resulting selection condition in several InfoPackages. Use a routine for time selection, if a simple interval or variables that already exist are not sufficient.

4.1.9.4.1.9 Tab Page: External Data

Use

You can use this tab page to specify from which file you load data.

Features

In addition to being able to make file-specific settings for the load process, you also have the option to preview and simulate loading from a file. This lets you check the data before you load it. From the preview, you can run a simulation of the data loading process. This allows you to check the update process.

This function makes it easier for you to check that the structure of the CSV and ASCII files you want to load is correct. It provides you with a better overview of data, particularly with hierarchies.

You can carry out the function after you have activated the transfer structure. To simulate the upload, you also have to create and activate update rules.

Preview

Once you have selected the file parameter information, the transfer structure is displayed in the preview as it appears after loading.

Simulation

The data loading process is simulated. Note that only the PSA transfer method is supported. With transaction data, the transfer rules and the update rules are simulated, and you can view the filled communication structure or the updated InfoCube. With attributes and texts, the transfer rules are simulated, allowing you to view the filled communication structure. With hierarchies, the hierarchy tree is displayed along with any error messages.

Activities

1. Specify whether the file from which you want to load data is stored on a workstation or on an application server.

   We recommend that you store the data on an application server, and load it from here into the BW. This is for performance reasons. This means that you can also load the data in the background.

2. Specify whether you want to load data from a data file or from a control file.

   A control file is an ASCII file that already contains the control instruction required by the scheduler. When you want to load a control data, you do not need to make any further entries, and the entered file is automatically controlled.

3. When loading data from a data file, make the following additional settings:

   a. Select the file to be loaded using the input help for the *File Name* field, or enter the file name directly, for example: **`C:/Daten/US/Kosten97.csv`**.

      You can also create a routine that determines the name of your file. If you do not create a routine for the name of the file, the system directly interprets the file name in the *File Name* field.

   b. If the file is not a control file, specify whether you want to load data from an ASCII file or a CSV file.

   – If you load data from an ASCII file, data is requested with a fixed data record length.

      If you want to upload a large amount of transaction data from a flat file and want to be able to specify the file type of the flat file, we recommend you create the flat file as an ASCI file.

      For performance reasons, uploading the data from an ASCII file is the most suitable method. Loading from a CSV file takes longer, because in this case the data separators and escape characters need to be sent and interpreted. However, generating an ASCII file might mean a larger workload under some circumstances.

– If you are loading data from an Excel CSV file, specify the *data separator* and the *escape character.*

Specify the separator that your file uses to divide the fields in the *Data Separator* field.

If the data separator character is a part of the value, the file indicates this by enclosing the value in particular start and end characters. Enter these start and end characters in the *Escape Characters* field.

You have chosen the ; character as a separator. However, your file contains the value **12;45** for a field. If you set **“** as the escape character, the value in the file must be **“12;45”** so that **12;45** is loaded into BI. The complete value that you want to transfer has to be enclosed by the escape characters.

If the escape characters do not enclose the value but are used within the value, the system interprets the escape characters as a normal part of the value. If you have specified **“** as the escape character, the value **12“45** is transferred as **12“45** and **12“45“** is transferred as **12“45“**.

In a text editor (for example, note pad) check the escape character and the separator currently being used in the file. These depend on the country version of the file you used.

Note that if you do not specify an escape character, the space character is interpreted as the escape character. We recommend that you use a different character as the escape character.

If you select the *Hex* characteristic, you can also specify the data separator and the escape character in hexadecimal format. When you enter a character for the data separator and the escape character, these are displayed as hexadecimal code after the entries have been checked. A two character entry for a data separator or an escape sign is always interpreted as a hexadecimal entry.

c. Specify additional parameters:

– *Thousand separator*

– *Character for decimal point*

– *Currency conversion for external systems*

– *Number of headers to be ignored*

4. Make the settings for the character record to be used.

5. To display the data before it is loaded or carry out a simulation of the loading process, choose with the quick info text *Preview*.

See also:

Maintaining InfoSources (Flat File) [Page 286]

Note the prerequisites for loading from CSV and ASCII files in this section.

4.1.9.4.2 Scheduling InfoPackages

Use

Determine the start conditions for the InfoPackage on the *Schedule* tab page.

Prerequisites

You have defined the conditions for data request on the scheduler tab page.

If you access an external system with the data request, you must make the data available in special formats. You can find additional information about this in the sections Maintaining InfoSources (Flat Files) [Page 286] and Maintaining InfoSources (External Systems) [Page 300]).

Procedure

1. Select whether you want to start the batch load process immediately or later on.

   Scheduling in the background is not possible for data requests from a client work station. In this case, you can only select *Start Data Load Immediately*. You can find additional information about loading flat files under Maintain InfoSource (Flat File) [Page 286] and under Tab Strip: External Data [Page 362].

   The initialization and execution of an InfoPackage is implemented as a process type in the process chain maintenance. The *Start Later in Batch* setting is hidden if the InfoPackage is used in a process chain. This is because the start of the request is determined by the process chain itself.

   If you do not use the InfoPackage in the process chain maintenance and you want to process the data request in the background, select the radio button *Start Later in Batch* and define the required start time. If start times already exist for this data request, this is indicated with a ✔ next to the *Scheduling Options* pushbutton. You can change the start time using *Settings Options*. You can also enter an end date for the data request with a background job.

   See also:

   Selection Conditions for the Start Date when Scheduling an InfoPackages in the Background. [Page 366]

   If you schedule an InfoPackage in the batch later, the background request job runs until all data is updated in BW. This is displayed with a corresponding indicator in the *Schedule* tab page. The load process remains active until the data is updated.

   For example, this gives external scheduling tools the option of monitoring the request job. The job then also runs with a request from an SAP source system as long as the source system has sent the data to the BW system and posted it or until the monitor sets the request to red. In the job log, a success or error message is displayed according to whether the request was posted successfully or not.

2. If you want to trigger an event that is to be used in the event collector after it was processed successfully or incorrectly, select: Postprocessing [External]. For more information on controlling processes using the event collector, see Process Control [External].

For controlling and monitoring BW processing as of release 3.0A, SAP recommends that you use Process Chains [Page 375].

If you are using an InfoPackage in a process chain, it can no longer be started in the background using the scheduler.

The *Postprocessing* setting is hidden when the InfoPackage is used in a process chain. This is because successor processes are set in the process chain, and not using the post processing events.

3. Select *Start* in order to start loading immediately or to schedule it.

Result

The load process is started either immediately or later in the background, depending on your settings. You can use the monitor [External] to check whether the load process was successful or not. After the load process is completed, successor processes are triggered if necessary, for example, when the InfoPackage was scheduled in a process chain.

Scheduled InfoPackages (and also InfoPackage groups) are denoted by a in the Administrator Workbench. You can delete or change the scheduled InfoPackages (and InfoPackage groups) at any time.

If you change already scheduled InfoPackages, your new selection is used in later background jobs, for example, periodic runs. You must create a new InfoPackage if you want to keep the existing InfoPackage, in order to continue using it in the background.

Note that some processes lock. See Process Lock Logic [External] for ODS objects and Process Lock Logic [External] for InfoCubes.

SAP recommends you serialize these processes using process chains.

4.1.9.4.2.1 Selection Criteria for Start Date when Scheduling in Background.

Use

The data request can be scheduled once or periodically. Here, periodically means that you determine the cycle in which the data is to be scheduled in the background job (hourly, daily, weekly, monthly, or over other periods). You have the option to start the data request immediately, on a particular date, at a particular time, after a job, after an event or in a particular operation mode.

Activities

One-off scheduling of the data request

1. Select the radio button *Start Data Load Immediately* or *Start Later in Batch.*

– You can use the *Scheduling Options* function to define the required start date.

– Save your entry of the *Start date values.*

You return to the initial screen *Tabstrip: Schedule*.

2. Start the job.

Periodic scheduling of the data request

1. Select the *Start Later in Batch* radio button.
2. Using the function *Scheduling Options*, you can define the required start date.
3. Set the indicator for *Execute Job Periodically* and select a period using the function *Period Values.*
4. Save your entries.

   You return to the initial screen *Tabstrip: Schedule*.
5. Select the runtime of the periodical job by selecting one of the radio buttons *Do not Terminate Job* or *Terminate Job after X Runs.*

   To prevent the system from endlessly requesting data, you can limit the data request with a numerical entry in the field *Terminate after X Runs*. The number you enter determines after how many runs the system is to terminate the load process.

See also:

The Background Processing [External] section in the documentation *SAP Web Application Server → Computing Center Management System (BC-CCM*).

4.1.9.4.3 Inserting an InfoPackage into the Process Chain

If you are using an InfoPackage in a process chain, the settings in the Scheduler are hidden for this InfoPackage, which are represented by a separate process type in process chain maintenance:

- Fully delete data target content
- Automatic further update from the PSA (before updating into data targets)
- Subsequent Processing
- Start later in batch
- Load hierarchy and mark for activation/activate

Any settings previously made in the InfoPackage are then ignored.

Use

If you want to automate the data load, you need to insert the InfoPackages in a process chain.

Prerequisites

You got to have created an InfoPackage already [Page 343].

Procedure

Call up the InfoPackage maintenance via *Process Chain Maintenance*	**Call up via *SAP Menu → Administration → Process Chains***
The *New Process Chain* dialog appears. 1. Enter the name and a description of the process chain you were wanting to create. Then choose *Next.* The dialog in which you can insert a start process into the process chain doth appear. 2. Create the start process. With master data and hierarchy, you can then get to the dialog which is used to insert an attribute change run into the process chain. 3. Where you see fit, insert the process into the process chain. The *Process Chain Maintenance Planning View* screen appears. Alongside the start process and any attribute change run (the case for master data and hierarchies), the system has inserted the load process and perhaps even the processes integrated with it, into the process chain.	The *Process Chain Maintenance Planning View* screen appears. 1. Navigate in the left-hand area in the chosen *Display Component* until you reach the *Process Chain*, into which you want to insert the load process. Select it with a double-click. The system displays the process chain plan view in the right-hand side of the screen. 2. If no suitable process chain is available, you need to create a new process chain. You can find additional information under Creating a Process Chain [Page 402]. 3. To insert a load process, in the left-hand area, choose *Process Types*. The system now displays the process categories available. 4. In the process category *Load Process and Postprocessing,* choose the process *Execute InfoPackage*. The dialog box for inserting a process variant appears. 5. Choose an InfoPackage. The system decides to transfer the InfoPackage into the process chain as well as the attribute change run (if we got master data and hierarchies), with the processes embedded with the load process where this is applicable.

Result

You have managed to insert the required InfoPackage into a process chain.

You might like to take advantage of some additional info on this subject under Process Chain Maintenance [Page 395] and Creating a Process Chain [Page 402].

4.1.9.4.4 Variants

Use

The creation of InfoPackages and their display in the Administrator Workbench are dependent on the variants.

A variant is user-defined grouping characteristic of the Scheduler. Every InfoPackage and every InfoPackage group is assigned to a variant. There are no empty variants, which means you must create a variant. The variant can be up to 12 characters in length and is case-sensitive. The system only recognises the variant when you have created and saved an InfoPackage (or an InfoPackage group) for the variant.

After saving a change in the Scheduler, the variant is also saved so that the variant is there for the user when he/she next calls up the Scheduler.

You cannot delete a variant.

The variant *Customer* is set as the default setting by the system.

However, you can switch variants or create a new one at any time in the Administrator Workbench, using *Settings* → *Change variant*...

Example

You can use the user name as a variant, or other names (of a department, for example)

4.1.9.4.5 Additional Scheduler Functions

- Testing delete selection at start.

  If you want to Automatically Delete or Cancel Requests for the InfoCube when loading [Page 357], via *Scheduler* → *Test Delete Selections at Start,* you can determine whether the requests are to be deleted or canceled with or without the system displaying a warning.

- Displaying initialization for a DataSource

  Under *Scheduler* → *Initialization Selection for Source System,* the system displays all initializations for the DataSource.

  You can find additional information from Tabpage: Updating [Page 358] in the **Initialization of Delta Process** section.

- Control Parameters Maintenance for Data Transfer [Page 370]

  With the settings made via *Scheduler* → *DataS. Default Values for Data Transfer*, you can override the settings you made in Customizing for a particular DataSource and a particular update mode.

- Indicating request as repair full request

  You can use a request that was selected as a repair request via *Scheduler* → *Repair Full Request* to carry out a full update in any data target. This also applies to data requests that already contain data from an initialization run or deltas for this DataSource/source system combination, and that have overlapping selection criteria.

  You can find additional information from Tabpage: Updating [Page 358] in the **Full Update** section.

- Setting time-out time

  You can use *Scheduler → Time-out Time* to set the maximum wait time after which the system determines the request as being in error. If the system has not received any (additional) data record before the maximum wait time has elapsed, the request assumes the status to be incorrect. You can use this function to override system-wide settings for a particular InfoPackage. The default setting for each InfoPackage is the same as the system-wide setting you made under *Monitor → Settings* (or in BW Customizing).

- Setting up warning handling

  Using *Scheduler → Warning Handling*, you can determine how the system reacts to warnings, should they occur, that is, whether the request gets the status green, yellow or red in such a case. You can use this function to override system-wide settings for a particular InfoPackage. The default setting for each InfoPackage is the same as the system-wide setting you made under *Monitor → Settings* (or in BW Customizing).

  We recommend you **do not** assign the status yellow to the request when a warning arises, since this prevents subsequent processing steps from starting. The request would then be denoted as incorrect after the time-out time had elapsed anyway.

.

4.1.9.4.5.1 Maintaining Data Transfer Control Parameters for DataSources

Use

In the BI Scheduler you have the option of determining control parameters for data transfer for individual DataSources. This function is available with *Scheduler → Default DataSource Settings for Data Transfer*. You can update transaction data in the PSA in this way, for example in larger data packages. Smaller packets guarantee faster processing if you want to update master data in the dialog.

You can define the following parameters for DataSources from SAP source systems:

Control Parameter	Update Mode
Size of data package	For all possible update modes
Number of parallel processes for the data transfer	For all possible update modes
Frequency with which the status IDocs are sent	For all possible update modes
Maximum number of data packages per request If there is an initial value, the system performs the data transfer according to the settings in the source system. A value 0 overwrites the setting in the source system. In this case there is no limit.	For delta and repeat requests

You can define the following parameters for file DataSources:

Control Parameter	Update Mode

Size of data package	For all possible update modes
Frequency with which the info IDocs are sent	For all possible update modes

Integration

DataSources from SAP Source Systems

The settings you made in the Scheduler override the settings made in extractor-customizing for SAP source systems for a particular DataSource and a particular update method.

In general you maintain the control parameters for data transfer from an SAP source system in the Customizing of the extractors. You go to Customizing of the extractors from the context menu of the corresponding source system in the source system tree of the Data Warehousing Workbench. Choose *Business Information Warehouse → General Settings → Maintaining Control Parameters for Data Transfer* to display or change the settings for the data transfer on source system level.

File DataSources

Use those settings that you made in the Scheduler to control the setting that you made under *Business Intelligence → Links To Other Systems → Maintain Control Parameters For Data Transfer* for a particular DataSource.

4.1.9.4.6 Handling Duplicate Data Records

DataSources for texts or attributes can transfer data records with the same key into BI in one request. Whether the DataSource transfers multiple data records with the same key in one request is a property of the DataSource. There may be cases in which you want to transfer multiple data records with the same key (referred to as duplicate data records below) to BI more than once within a request; this is not always an error. BI provides functions to handle duplicate data records so that you can accommodate this.

Dataflow with Transformation

In a dataflow that is modeled using a transformation, you can work with duplicate data records for time-dependent and time-independent attributes and texts.

If you are updating attributes or texts from a DataSource to an InfoObject using a DTP, you can specify how you want the system to handle data records that have the same record key. In DTP maintenance on the *Update* tab page, you set the *Handle Duplicate Record Keys* indicator.

This indicator is not set by default.

If you set the indicator, duplicate data records (multiple records with identical key values) are handled as follows:

- Time-independent data:

  If data records have the same key, the last data record in the data package is interpreted as being valid and is updated to the target.

- Time-dependent data

  If data records have the same key, the system calculates new time intervals for the data record values. The system calculates new time intervals on the basis of the intersecting time intervals and the sequence of the data records.

Data record 1 is valid from 01.01.2006 to 31.12.2006

Data record 2 has the same key but is valid from 01.07.2006 to 31.12.2007

The system corrects the time interval for data record 1 to 01.01.2006 to 30.06.2006. As of 01.07.2006, the next data record in the data package (data record 2) is valid.

If you set the indicator for time-dependent data, note the following:

You cannot include the data source field that contains the DATETO information in the semantic key of the DTP. This may cause duplicate data records to be sorted incorrectly and time intervals to be incorrectly calculated.

The semantic key specifies the structure of the data packages that are read from the source.

Example

You have two data records with the same key within one data package.

In the following graphic, DATETO is not an element of the key:

	Key	DATETO	DATEFROM	Value
Rec. 1	A	31.12.2006	01.01.2001	xyz
Rec. 2	A	31.12.2001	01.01.2000	abc

In the data package, the data records are in sequence DS2, DS1. In this case, the time interval for data record 1 is corrected:

Data record 1 is valid from 1.1.2002 to 31.12.2006.

Data record 2 is valid from 1.1.2000 to 31.12.2001.

In the following graphic, DATETO is an element of the key:

	Key		DATEFROM	Value
		DATETO		
Rec. 1	A	31.12.2006	01.01.2001	xyz
Rec.2	A	31.12.2001	01.01.2000	abc

If DATETO is an element of the key, the records are sorted by DATETO. In this case, the data record with the earliest date is put before the data record with the most recent date. In the data package, the data records are in sequence DS2, DS1. In this case, the time interval for data record 2 is corrected:

Data record 2 is valid from 1.1.2000 to 31.12.2000.

Data record 1 is valid from 1.1.2001 to 31.12.2006.

If you do not set this indicator, data records that have the same key are written to the error stack of the DTP.

Dataflow with 3.x Objects

In a dataflow that is modeled using 3.x objects, you can only work with duplicate data records for time-independent attributes and texts.

If the DataSource transfers potentially duplicate data records, this information is passed to the scheduler when you create new InfoPackages. On the *Processing* tab page, the *DataSource Transfers Duplicate Data Records* indicator is set. The *Ignore Duplicate Data Records* indicator is also set. When multiple data records that have the same key are transferred, the last data record in the request is updated to BI. Any other data records in the request with the same key are ignored.

To maintain consistency, ignoring duplicate data records is only possible if the data is updated serially. A serial update is when data is first updated into the PSA then, after it has been successfully written to the PSA, it is updated into the master data or text tables of the InfoObject.

If a DataSource transfers potentially duplicate data records or if you manually set the *Ignore Duplicate Data Records* indicator, the *PSA Only* update type is automatically selected in the scheduler.

If you deselect the *Ignore Duplicate Data Records* indicator, any duplicate data records cause errors. The error message is displayed in the extraction monitor.

Since the characteristic key is used to identify duplicate data records, make sure that all the key fields of a characteristic are filled by the DataSource. If this is not the case, for example because key fields are hidden in the extract structure, the system notifies you of this when you activate the transfer rules.

Note

You can specify how duplicate data records within a request are handled, independently of whether the setting that allows DataSources to deliver potentially duplicate data records has been made. This is useful if the setting is made for the DataSource but the system knows from other sources that duplicate data records are transferred (when loading flat files, for example).

4.1.9.4.7 Handling of Data Records with Errors

Use

Using the *Error Handling* function on the *Update Parameters* tab page in the scheduler, you can control the behavior of the BI system if data records with errors [External] occur in the data flow with 3.x objects (use of transfer and update rules) when you are loading data using the PSA.

The system checks for data records with errors:

- in the transfer rules
- in the update rules
- when data is updated to the master data tables, text tables, or hierarchy tables
- when data is updated to the InfoCube
- when checking for referential integrity [Page 279] within an InfoObject against the master data tables or DataStore objects on the communication structure level.

Features

Incorrect records arising in transfer rules, update rules, when updating to a data target, or when checking for referential integrity, are highlighted and returned to the error handler. The error handler locates the original PSA records and writes the corresponding log. This function is not yet available for updating to DataStore objects.

Using the *Error Handling* function on the *Update Parameters* tab page in the scheduler, you can control how the system responds to incorrect data records.

Activities

1. Define how the system should react to incorrect data records

   a. No update, no reporting (default)

      If errors occur, the system terminates the update of the entire data package. The request is not released for reporting. However, the system continues to check the records.

   b. Update valid records, no reporting (request red)

      This option allows you to update valid data. This data is only released for reporting after the administrator checks the incorrect records that have not been updated and manually releases the request (using a QM action, that is, by setting the *Overall* status on the *Status* tab page in the monitor).

   c. Update valid records, reporting possible

      Valid records can be reported immediately. Automatic follow-up actions, such as adjusting the aggregates, are also carried out.

      With option a), the incorrect records are marked in red in the PSA maintenance. You can display the relevant error messages amodally, edit the records, and update the request manually. If it was not possible to write to the PSA (hierarchy or transfer method IDoc), an application log is written instead.

      With options b) and c), a new request that is only updated to the PSA is formed from the incorrect records. Here you can edit the records of the new request in the PSA and start the update manually.

      If errors arise when an InfoCube is updated, new requests are generated for incorrect records for each InfoCube combination. Among these requests, the incorrect records are updated to the PSA. These new requests have entered as data targets the InfoCubes for which there were incorrect records. If, for example, a record was updated to two InfoCubes by mistake, a request is generated for this record that contains both InfoCubes as data targets. In the PSA tree, the requests appear as successfully updated in the PSA. These new requests can then be triggered again for updating to the InfoCubes.

2. Specify the maximum number of incorrect data records that are allowed before the system terminates the load process. In the PSA, you can display the errors that have arisen up to that point. If you do not specify the maximum number of incorrect records after which termination should occur, the handling of incorrect data records is inactive and the update terminates after the first error.

3. Define how the BI system should react if the number of data records received by the BI system is not the same as the number of data records updated in BI.

   A difference between the number of received and updated records can occur:

   - in the transfer rules

- in the update rules
- when updating

Records are sorted, aggregated, or added.

If you set the *No Aggregation Permitted* indicator, the request is deemed incorrect if the number of received records is not the same as the number of updated records.

Regardless of this indicator, an error occurs if the number of selected records is not the same as the number of records received in the BI system.

Result

The monitor directs the user to the relevant error case and enables a direct navigation into the PSA or the respective error request.

If you want to continue processing the incorrect data records in a program that you have written yourself, you can use the method GET_ERRORS of class CL_RSSM_ERROR_HANDLER. The use of this method is documented in the RS_ERRORLOG_EXAMPLE.

More Information:

Checking and Changing Data Using the PSA APIs [Page 52]

4.1.10 Process Chain

Definition

A process chain is a sequence of processes [Page 377] that are scheduled to wait in the background for an event. Some of these processes trigger a separate event that can, in turn, start other processes.

Use

In an operating BI system there are a multitude of processes that occur regularly. If you use process chains, you can:

- **Automate** the complex schedules in BW with the help of the event-controlled processing,
- **Visualize** the processes by using network graphics, and
- **Centrally control and monitor** the processes.

Fundamental principals of the process chain concept are:

- Openness

  The abstract meaning of a process as any process with a defined beginning and end enables openness with regard to the type of process that can be integrated into a process chain. The principle of openness is applied to the theory behind process chains, in that both user-defined programs and processes can be implemented. In addition, you can include process chains in other process chains, known as meta chains. In doing so you are able to integrate process chains from the system in which the meta chain is found, or from other systems. In this context, we are talking about local or remote process chains.

- Security

  Using process chains offers a high amount of process security, which is based on the principals of background management:

  – Processes are scheduled before they run and can be monitored with the standard batch monitor.

  See also: Process Chain Log Display [Page 412]

  – Background events start subsequent processes.

  – Short dumps and terminations are recognized and handled respectively.

- Flexibility

  The subsequent process must get all the information it needs for a correct run from its predecessors. This allows new process types to be integrated without the existing types having to be adjusted.

Structure

A process chain consists of a start process, individual application processes and the collection processes.

Define the start of your process chain with the start process [Page 379]. All other chain processes are scheduled to wait for an event.

The application processes [Page 390] are the actual processes. BI supports process types of the following categories:

- General services
- Load process and post processing processes
- Data target administration processes,
- Reporting Agent processes
- Other BI processes,

as well as processes that you have implemented.

If they are used in other SAP applications, you have further categories available, if applicable.

Collection processes [Page 379] are treated differently by the process chain management. They allow multiple chain strings to be combined to form one individual string. This allows them to replace multi-field scheduling of the actual work processes.

Processes are connected using events that start a successor process after being triggered by a predecessor process.

Integration

A process chain is a BI object with a transport connection [External] and a connection to the BI document management [External].

Automatisms

If you use process chains, the automatisms of the integrated processes (for example, update PSA data in the data target, or activate data in the DataStore object) are ignored and you must implement them using the process chain. If you schedule a specific process in a chain, you support the automatic insertion of additional, relevant standard processes with the consideration of such automatisms.

If you use data transfer processes, the automatisms from InfoPackages are no longer available and you must implement them using process types.

4.1.10.1 rocess

Definition

A process in the context of process chains is a procedure inside of or external to an SAP system with a defined beginning and end.

A distinction is made between a start process, an application process and a collection process.

Use

In order to automate runs in BW, processes are brought together in so-called process chains. The processes are scheduled in the background. Each process can trigger one or more events, that trigger other processes in turn.

Structure

A process is characterized by

- the process type

  The process type is the kind of process, for example load process. The process type decides which tasks the process has and which properties it has in the maintenance.

  The process type is set in the RSPROCESSTYPES view.

  See also: Maintenance of Process Types [Page 415]

- the process variant

  The process variant is the name of the process. Within the process chain context, it displays the configuration of a process of a particular type set at the time of definition. A variant is only consistently defined together with the process type. A process can have various variants. During the load process, for example, an InfoPackage represents a process variant.

  The user defines the process variant at the time when the process is scheduled. With some process types, the variants are determined internally and are stored as GUIDs.

- the process instance

  The process instance is the characteristic value of the process It contains the most important information that the process, or subsequent processes want(s) to communicate. In the load process, this would be the name of the request for example. If the process is ended, the instance is transferred to the process chain management and saved. The logs for the process are stored under the process instance.

  The instance is determined by the process itself at run time, and is normally determined consistently, independent of time or system.

4.1.10.2 Serviceprozesse

Definition

Beantworten Sie die folgende Frage: Wie ist das Objekt definiert?

Löschen Sie anschließend die Absätze in grüner Schrift.

Verwendung

Dieser Abschnitt ist fakultativ.

Beantworten Sie die folgenden Fragen:

- Wie kann der Kunde dieses Objekt allgemein verwenden?
- Wo kann der Kunde dieses Objekt verwenden?
- Welche Einschränkungen gelten für die Verwendung des Objekts?

Löschen Sie anschließend die Absätze in grüner Schrift.

Struktur

Dieser Abschnitt ist fakultativ.

Beantworten Sie die folgenden Fragen:

- Aus welchen Teilen besteht das Objekt?
- Welchen Zweck erfüllt jedes Einzelteil?

Löschen Sie anschließend die Absätze in grüner Schrift.

Integration

Dieser Abschnitt ist fakultativ.

Beantworten Sie die folgende Frage: Welche Beziehungen bestehen zwischen diesem Objekt und anderen Objekten?

Löschen Sie anschließend die Absätze in grüner Schrift.

Beispiel

Dieser Abschnitt ist fakultativ.

Beantworten Sie die folgende Frage: Wie sieht ein Beispiel für dieses Objekt aus?

Löschen Sie anschließend die Absätze in grüner Schrift.

4.1.10.2.1 tart Process

Definition

You can define the start of a process chain with the start process.

Use

The background control options are available to **directly schedule** the start process. You can start the process chain immediately (when activating the process chain), at a specified time, or after a particular event. When you activate the process chain the start process is scheduled in the background, as defined in your selections.

If there are not sufficient options available, you can trigger the start of the process chain **using API**. You can use the SAP NetWeaver Scheduling Framework to start the chain and to have more extensive scheduling options.

You can also trigger the start of a process chain **using a metachain.** A metachain is a process chain, for which you determine this start condition, that is fixed to another process chain. The process chain is started directly by this metachain.

If you start the start process using a metachain, it is not scheduled after you have activated the related process chain. The process chain is only started when the metachain, to which it is linked, is running.

The remaining chain processes; the application processes and the collection processes, are scheduled to wait for an event.

The start process has the following special features:

- Only the start process can be scheduled without a predecessor process.
- The start process cannot be a successor of another process.
- Only one start process is allowed for each process chain.
- A start process can only be used in a single process chain.

4.1.10.2.2 ollection Process

Definition

A collection process collects several chain strings to form one string in the process chain maintenance.

Use

Process chain management handles collection processes in a particular way. The system makes the variant names consistent and guarantees that all processes of the same name that have been scheduled more than once, trigger the same event. This enables the several chain strings to be collected to form one and also makes multiple-scheduling of the actual application processes unnecessary.

The following collection processes are available in the process chain maintenance:

- And Process (Last)

  This process does not start before all events of the predecessor processes, that is including the **last** event, that it has waited for, have been successfully triggered.

Use this collection process when you want to combine processes and when further processing is dependent on all these predecessors.

- Or Process (Every)

  The application process starts **every time** a predecessor process event has been successfully triggered.

  Use this collection process when you want to avoid multi-scheduling the actual application process.

- XOR Process (First)

  The application process starts when the **first** event in one of the predecessor processes has been successfully triggered .

  Use this collection process when you want to process processes in parallel and schedule further independent processes after these ones.

  Owing to the time components, the collection processes do not display logical gates in the normal sense, because the system cannot distinguish between whether a 0 entry (**no event received**), means that the event was **never received** or whether it has **not been received yet**. The reason for this is that the checks do not run continuously but only take place when the event is received.

4.1.10.2.3 Interrupt-Prozess

Definition

Der Interrupt-Prozess ist ein Prozesstyp, mit dem die Abarbeitung einer Prozesskette bis zur Erfüllung einer Bedingung unterbrochen werden kann.

Verwendung

Wenn eine Prozesskette im Ganzen oder in Teilen nur abgearbeitet werden soll, falls mehr als eine Startbedingung erfüllt ist, so benutzen Sie diesen Prozesstyp, um die zusätzlichen Bedingungen anzugeben.

Wie beim Startprozess stehen Ihnen die Optionen der Hintergrundsteuerung zur Verfügung, z.B. Sofortstart, Start zu einem bestimmten Zeitpunkt oder nach einem bestimmten Ereignis, und Sie können festlegen, ob die Startbedingung periodisch sein soll.

Integration

Eine Prozesskette wird bei Verwendung des Interrupt-Prozesses folgendermaßen ablaufen: Die Kette wird starten, wenn die Bedingung des Startprozesses erfüllt ist. Der Interrupt-Prozess wird jedoch die Abarbeitung der Kette unterbrechen, bis die Bedingung des Interrupt-Prozesses erfüllt ist. Dabei behält er den Status *aktiv*.

Der Interrupt-Pozess plant einen weiteren Hintergrundjob ein, welcher auf die jeweilige Bedingung hin startet. Daher ist der Interrupt-Prozess während der Unterbrechungsphase nicht wirklich aktiv, es werden währenddessen keine Ressourcen verbraucht.

Wenn die Bedingung des Startprozesses ein weiteres Mal erfüllt ist, bevor die Bedingung des Interrupt-Prozesses erfüllt ist, so wird die Kette ein weiteres Mal starten und ebenfalls nur bis

zum Interrupt durchgeführt werden. Sobald dann die Bedingung des Interrupt-Prozesses erfüllt ist, wird nur der letzte Lauf der Kette fortgesetzt. Die früheren Läufe bleiben unverändert.

Wenn die Bedingung des Interrupt-Prozesses vor der Bedingung des Startprozesses erfüllt ist, so wird die Kette starten, sobald die Startbedingung erfüllt ist, der Interrupt-Prozess wird die Kette jedoch nicht mehr unterbrechen, da dessen Bedingung bereits vorher erfüllt war. Wenn die Bedingung des Interrupt-Prozesses ein weiteres Mal erfüllt ist, bevor die Startbedingung der Kette erfüllt ist, so hat das keinen Einfluss auf die Abarbeitung der Kette, sie wird auch dann durch den Interrupt-Prozess nicht aufgehalten.

Abhängigkeiten

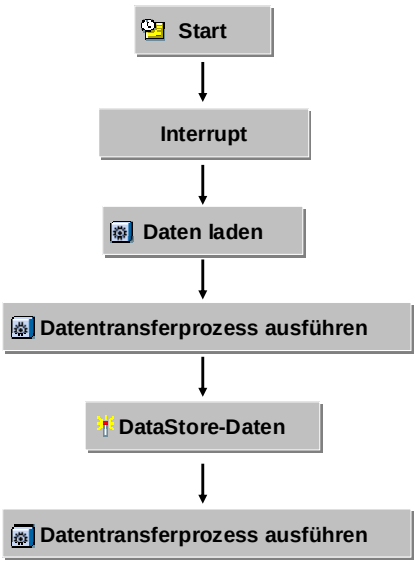
Beachten Sie, dass die Kette den Status *aktiv* behält, bis der Interrupt-Prozess abgelaufen ist. Daher eignet sich dieser Prozesstyp nicht, um bestimmte Äste der Kette von der Abarbeitung auszuschließen. Hierfür verwenden Sie den Prozesstyp Bedingung/Entscheidung?.

Der Interrupt-Prozess ermöglicht die UND-Verknüpfung von Startbedingungen einer Prozesskette. Wenn Sie ODER-Verknüpfungen benötigen (z.B. "Kette startet montags oder wenn ein anderer Prozess beendet ist"), so können Sie dies über Meta-Ketten modellieren, von denen jede auf je eine der Bedingungen eingeplant ist, und welche ihrerseits die eigentliche Kette startet.

Eine Metakette mit Start Sofort und zwei Unterketten, die nach Startbedingungen starten (nicht über Meta-Kette). Unterketten mit ODER verbunden, um Nachfolgeprozess zu starten?

Beispiel

Im folgenden Beispiel werden die Daten erst nach Eintreten der Startbedingung (z.B. dem Eintreten eines bestimmten Systemereignisses) sowie nach Eintreten einer weiteren Bedingung (z.B. einer bestimmten Uhrzeit) geladen und anschließend in ein DataStore-Objekt und in einen weiteren InfoProvider fortgeschrieben:



4.1.10.2.4 ABAP Programs

Definition

Within the context of process chain maintenance, an ABAP program is a process type.

Use

Use the **ABAP Program** process type if you want to use a simple, independent program in a chain or if you want to use a program scheduled by a user or another program in the background. This allows you to schedule a program of your choice (ABAP report) in the process chain management, with or without **program** variants.

The scheduling options are as follows:

- Synchronous, local

  Specify the program name and variant under *Program to be Called → Program.* The program is started after the event for the predecessor process has been triggered.

  The program is executed in the background on the same server on which the process chain is scheduled. The process chain waits until the program has finished before it continues with any possible successors.

- Asynchronous

  Choose this option if your program is asynchronous.

  An asynchronous process is also called a distributed process. Distributed processes are characterized as having different work processes involved in specific tasks. A program is asynchronous when it does not end the task which it started itself. In an asynchronous process, the process chain management does not equate the end of the process with the end of the background process. The process status remains **active.**

  In asynchronous processes, successor events are not automatically triggered.
  You need to call the remote-enabled `RSPC_ABAP_FINISH` function module at the end of your program to signify the real end of your process. The process is then indicated as completed and the successor events are triggered.

  Inform the function module of the **process** variant, with which your program is scheduled. To do this, you need to either hardcode the process variant in the program, or transfer the **process** variant to a **program** variant You need the following lines in your program to do this:

  ```
  PARAMETERS variant TYPE rspc_variant.
  ```

  The process chain management does not recognize the termination of your program.

  Therefore, when you restart the chain it will also terminate, since the old run is not finished. To avoid this, you need to manually set the process to *terminated* in the chain log view in the process monitor (*Process* tab page). Do this prior to restarting.

- Asynchronous, local

  Choose this option if your program is asynchronous (it effects a user decision or makes an asynchronous RFC); or if you want to start a scheduled program in the background. In this case, the program must be scheduled to wait for a batch

event. Enter the event and, if necessary, the parameter under *Program to be Called → Scheduled Program.*

– Asynchronous, destination

In this option you must schedule the program to be called on the target server to wait for an event.

Under *Called From,* enter the *Destination* for the target system and enter the event and, if necessary, the parameter by choosing *Program to Be Called → Scheduled Program.*

This option is only available for target systems as of service API 3.0A.

We recommend that you write a local program, in which you synchronously call a remote-enabled function module in the target system. Then choose the option *Synchronous, Local.* This allows you to monitor your process using process chain management.

You can write logs without much effort by issuing success messages. These logs are displayed, using the context menu for a process, in the process chain maintenance log view. Choose *Display Messages→ Tab Page Batch.*

Error messages terminate the job. Therefore, write your logs as success messages. Furthermore, for this process type, it is not possible to evaluate the error messages as a status in the program so that you can trigger possible success processes *By Error.* If you require this function, see Implementing Your Own Process Type [Page 393].

Integration

We recommend you model more complex processes not covered by BI process types, using Implementing Your Own Process Type [Page 393].

4.1.10.2.5 perating System Command

Definition

Within the context of process chain maintenance, a operating system command is a process type.

Use

You can use this process type to place an external command in an application server's operating system.

See also:

Documentation External Commands and External Programs [External]

After the run has completed, you can find the text produced by the command as a standard output by accessing the *Process* tab page of the log for this process.

For processes you have implemented yourself, this log can be called up using the method CL_RSPC_SYSTEMCOMMAND⇒IF_RSPC_GET_LOG~GET_LOG.

See also:

Process Chain Logs Display [Page 412]

4.1.10.2.5.1 Including Operating System Commands as Processes in Process Chains

Procedure

You are in the plan view of the process chain, in which you want to include an operating system command as a process.

The operating system command process type is available in the *General Services* process category.

1. Insert the operating system command process type into the process chain using Drag&Drop.
2. In the next dialog box, select a process variant and confirm your entry.

   To create a new process variant, choose *Create*.

   a. In the resulting dialog box, enter the description of the process variant and choose *Next*.

      The maintenance screen of process variants for an operating system command appears.

   b. Here, under *Execution*, define:

   – The command name

      The command is defined using *CCMS → Configuration → External Commands.* You can also create and change a command using the *Change* function.

      For more information, see Defining External Commands [External].

   – The operating system of the application server, on which you want to execute the command.

   – Additional parameters, providing these are allowed by the defined command.

      If a user is allowed to specify additional arguments when scheduling the external command in a background job, you can specify these arguments here. The additional parameters are added to the parameter strings specified in the command definition, under *Parameters for the Operating System Command.*

   – The application server on which the command is to be executed.

      If you always want to use the current host, set the corresponding indicator. If you enter a different host to the current one in the *Target Machine (Host)* field, the system automatically deselects the *Current Host* indicator.

   c. Under *Evaluation*, set whether you want to evaluate the output of the command.

      You can evaluate the process as either successful or having errors if the output of the command contains a character string that you specified. If the system cannot find the character string, the process is either ended by the counter status, or repeated. When you have determined that the process is to be

repeated, you can specify after how many seconds and how often it is to be repeated.

This allows you to check, for example, whether a particular file exists in a directory before you load data to it. If the file is not available in the directory, the check is repeated according to the previously specified intervals before the process is ended and loading starts.

d. Save your entries and go back.

e. In the following dialog box, confirm the insertion of the process variants into the process chain.

Result

You have included an operating system command as a process in a process chain. After including any additional process variants in your chain, check the chain and then activate it.

4.1.10.2.6 Local Process Chains

Definition

Process chains can be scheduled as processes in additional process chains. These are called metachains. If the process chain that you want to execute is in the metachain, use the *Local Process Chain* process type.

A local process chain consists of processes that are scheduled in the same system as the metachain.

Use

You use a local process chain to schedule a process chain as a process of a metachain.

Integration

In addition to local process chains, you can also schedule remote process chains as processes in a metachain. A remote process chain calls a process chain in another system. This is executed in the other system.

See also:

Start Process [Page 379]

4.1.10.2.7 Process Chain Remote

Definition

Process chains can be scheduled as processes in additional process chains, which are known as meta chains. If the process chain you want to execute is in another system, use the process type *Process Chain Remote,* called remote process chain in the following. A remote process chain is a local process in the calling system that calls a process chain in this system,

using a destination in another system, that is to be executed there. The remote process chain communicates with the other system and synchronously starts the executed process chain. After terminating the process chain in another system, this is reported back to the meta chain in the calling system.

Prerequisites

Before you can run a process chain in another system, you have to execute the following steps:

- In the system to be called, define the process chain that is to be run there.
- Maintain a destination for the system to be called.
- Maintain a destination, that is displayed back in your calling system, in the system to be called.

Use

Using a remote process chain as the process of a metachain, you can schedule a process chain from another system in a process chain of the calling system.

Integration

In addition to remote process chains, you can also schedule local process chains as processes in a metachain. A local process chain consists of processes that are scheduled in the same system as the metachain. You schedule it as a process in the same system as the metachain.

4.1.10.2.8 Workflow

Definition

In the context of process chain maintenance, a Workflow [External] can be both a process type and also a worklow activity.

4.1.10.2.8.1 Including a Workflow as a Process in a Process Chain

Prerequisites

In the system, in which the workflow is executed, you have carried out automatic workflow customizing (transaction SWU3).

The system in which the workflow is to be executed has Basis release 4.0B or higher and BW Service API 3.0B as a minimum.

If you want to end the workflow with a more complicated status (with multi-value end options), the system in which the workflow is executed must have a release level from SAP NetWeaver 2004s.

Requests Sent to the Workflow to Be Executed

A model workflow has been delivered to display requests sent to the workflow. The workflow model is : WS43200002 (ropc_verify2).

The most important task that the workflow must contain is dispatching the end notification to the process chain. This task was also delivered as standard task TS43200003. Note how the container element “&CALLINGCHAIN&” (business object “ROPE_WF”) is bound to the work item ID.

In the delivered example, another user decision was inserted to represent the *Miscellaneous Tasks* of the workflow.

Procedure

You are in the plan view of the process chain, in which you want to include an workflow as a process.

The workflow process type is available in the *General Services* process category.

1. Insert the process into the process chain with Drag&Drop.
2. In the next dialog box, select a process variant and confirm your entry.
   a. To create a new process variant, choose *Create*.
   b. In the resulting dialog boxes, enter the technical name and a description of the process variant and choose *Next*.

      The maintenance of process variants for a workflow appears.
   c. Choose the destination in which the workflow is to be executed.

      In the destination into the remote system, use the usual background user, so that the assigned profile S_BI_WX_RFC has the authorizations necessary for executing and monitoring a workflow. Note: However, this destination is also used for dialog logons, for example, for workflow monitoring.

      If you do not specify the destination, the specified workflow is executed in the system directly.
   d. Choose the workflow to be executed (also called model workflow, or multi-step task).
   e. You can also create and change a workflow with the *Change* function.
   f. If required, choose a destination in the remote system with which the workflow can call the process chain back.
   g. If required, specify the end option which is to be transferred to the process chain. The relevant event is triggered, based on the options entered here.

      If you do not specify an option, the process ends successfully without an option.

      For more information about setting end-options for a process, see Maintenance of Process Types [Page 415].
   h. Save your entries and go back.
   i. In the resulting dialog box, confirm the insertion of the process variant into your process chain.

Result

You have inserted a workflow as a process into a process chain. After including any additional process variants into your chain, check the chain and then activate it.

4.1.10.2.8.2 Executing a Process Chain Within a Workflow

Prerequisites

If you want to execute a process chain in a remote system, you must connect the starting system in which the workflow is executed to the BW system as a source system in which the process chain is executed. This is because the callback destination is returned to the workflow system using the relationship to the source system.

Requests Sent to the Process Chain

- You have activated the process chain and it is scheduled in the background.
- In the start process of the process chain to be executed, *Start with Metachain or API* is set.

Requests Sent to the Workflow

A model workflow has been delivered to display requests sent to the workflow. The workflow model is: WS43200001 (ropc_verify1).

The workflow must contain two activities:

1. Starting the process chain

   This task was also delivered as standard task TS43200001. This task is a “creation activity”. The process chain to be started as well as the destination on which the process chain can be found is supplied with the ROPC process chain object to be created. The task receives the process chain instance as a return.

   Note the binding of “&CHAIN&“ and “&DESTINATION&“ to the export parameters “Chain“ and “Destination“ and the return “&PROCESSCHAINSAPI&“.

2. Wait until the process chain has finished.

   This is a wait step that ends when the process chain triggers the event.

   Ensure that business object ROPC is the container element.

In the delivered example, an additional task, “displaying the process chain” is supplied. This shows you how to access the process chain log from the workflow.

Procedure

See documentation related to the Workflow Builder [External] in the documentation of the SAP Web Application Server, under *Business Management → WebFlow Engine → Reference Documentation → SAP Business Workflow – Navigation*.

4.1.10.2.9 Decisions

Definition

A process type, in the context of process chains.

Use

The *Decision* process type allows you to determine a set of conditions which form the basis of a decision, in accordance with the following logic:

“If condition A is filled, option X applies, if condition B is filled, option Y applies,... otherwise, option Z applies“

The conditions have a description and are formulated as logical expressions using the formula builder.

In the formula builder, only syntax fields are directly available as fields. In particular, this allows decisions on the basis of date and time. To incorporate this, the function library in the formula builder has been enhanced with the formula function `DATE_WEEKDAY1`.

The DATE_WEEKDAY1 formula function calculates the day of the week as a technical specification (1..7) from the date. For example, you can check the form DATE_WEEKDAY1(date of the weekday) = '6' OR DATE_WEEKDAY1(date of the weekday) = '7'. This allows you to change the way the chain runs at the weekend, in comparison to the other days of the week.

The function library of the formula builder has also been enhanced by the *Process Chain* category. This allows you to access additional information within the formula. The *Process Chain* category contains the following formula functions:

- PREDECESSOR_PARAMETER: Runtime parameter of direct predecessor
- PROCESS_PARAMETER: Runtime parameter of a process in the current chain
- PROCESS_VALUE_EXISTS: Process in chain has parameter value

These functions allow you to access values from predecessor (and successor) processes of the current process chain. If the process is a predecessor process, the system requests the instance values; otherwise the maintained variant values are requested.

4.1.10.2.9.1 Including Decisions as a Process in Process Chains

Procedure

You are in the plan view of the process chain, in which you want to include a decision as a process.

The *Decision* process type is available in the *General Services* process category.

1. Insert the *Decision* process type into the process chain by using drag and drop or by double-clicking.
2. In the next dialog box, select a process variant and confirm your entry.

   To create a new process variant, enter a name for the variant and choose *Create*.

   a. In the following dialog box, enter the description of the process variant and choose *Next*.

   b. The maintenance screen of process variants for a decision appears.

      Enter the decision paths for the process variant in the table; that is, you specify the conditions on the basis of which specific options are to be selected as events.

      For a decision path, the table displays in a row the condition that has to be fulfilled as well as the event that is triggered when fulfilling a condition.

When creating the variant, the system displays the first row with the *if … then* commands and the second, last row with the commands *otherwise … then*. You can insert additional rows. These are inserted between the first and last rows with the *otherwise if … then* commands.

Specify the condition in the *Formula* column using the formula builder for each row by choosing *Create*. For the formula, enter the description that is displayed in the table.

For more information about using the formula builder, choose with the quick info text *Information* in the application toolbar of the formula builder.

When using the *process chains* category function in the formula builder, note the following:

Input help is available. The input help refers to the process type. Fill the function parameters from left to right with the *Constant* pushbutton. The dialog box that appears enables input help that is dependent on the current process chain and the parameters for the function that have been maintained.

Additionally, specify in each row which option is to be chosen when fulfilling the condition as an event.

For the last row, you cannot specify a formula because the condition is “If none of the previous conditions is fulfilled, chose <Option X> as event.”

c. Save your entries and return to the previous screen.

d. In the following dialog box, confirm the addition of the process variants in the process chain.

3. Link the process to the follow-up processes. The dialog box for determining the color of the connection allows you to select one of the options you defined and the connection when there are errors.

Whenever a follow-process waits for an option, follow-up processes have to be available for the other possible options because a possible follow-up process that is missing leads to errors when the missing option is chosen. If this is not the case, the system outputs a warning during the scheduling time check. On the other hand, the system outputs a warning when a follow-up process is waiting for an option that is not declared.

Result

You have included a decision process as a process in a process chain. After including any additional process variants in your chain, check the chain and activate it.

4.1.10.3 pplication Process

Definition

Application processes are processes that you want to automate in process chain maintenance. They represent activities typically performed in the operative use of BI.

Application processes supported by process chain maintenance

Process Category	Application Process Type
Load process and postprocessing	• Execute InfoPackage For more information, see Including InfoPackages in Process Chains [Page 367]. If you are using process chains, the automatisms of the integrated processes are ignored and have to be implemented using the process chain. For example, in the InfoPackage on the *Update* tab page, you select update types *Only PSA* and *Subsequently in Data Targets*. You are using this InfoPackage in a process chain. A further update into the data target does **not** run automatically. This is performed after the load process when the *Read PSA and Update Data Target* process is scheduled in the process chain. Reading the PSA and Updating Data Targets [Page 55] • Save hierarchy See Loading Hierarchies Using a Process Chain [External] → Step 7. Updating DataStore object data [External] (Further Update) • Export data to non SAP systems See Including InfoSpokes in Process Chains [External] and Creating InfoSpokes [External]. Deleting Overlapping Requests from an InfoCube [Page 357] • Trigger event data change (for broadcaster) See also Including Event Data Changes in Process Chains [External].

Data target administration	• Delete index See InfoCube Performance [Page 736] → Deleting Indexes section and Including InfoCubes in Process Chains [External]. • Build index See Including InfoCubes in Process Chains [External]. • Build database statistics See InfoCube Performance [Page 736] → Database Statistics section. Filling New Aggregates [Page 684] Rolling up Filled Aggregates [Page 687] Compressing the InfoCube [Page 735] Activating ODS Object Data [External] • Delete data target contents in full
Reporting agent	• Exception reporting • Printing in the background • Precalculation of Web templates • Precalculation of value sets For more information, see Including Packages in a Process Chain [External] in the Reporting Agent documentation.
Other BI processes	System Response Upon Changes to Master Data and Hierarchies [Page 689] • Adjusting time-dependent aggregates For more information, see Editing Aggregates Manually [Page 672] → step 8. Deleting Requests from the PSA [Page 44] Reorganizing Master Data Attributes and Texts [Page 123]
General services	ABAP Program [Page 382] Operating System Command [Page 383] Local Process Chain [Page 385] Remote Process Chain [Page 385] Workflow [Page 386] (also remote)
	Implementing Your Own Processes [Page 393]

In the system, you can call up documentation for a process type by selecting the process type in the overview tree and choosing F1. You find the overview tree in process chain maintenance on the left-hand area of the screen. You can also call the documentation from

the dialog box in which you specify variants. You do this by choosing *Further Help* when you add a process to the process chain.

4.1.10.4 Implementing Your Own Process Type

Purpose

If you:

- write your own application log to be displayed in the *Process* tab strip for the Logs for a Process in the Process Chain Maintenance [Page 412], or:
- if you have implemented your own monitor, or
- have implemented a Customizing interface that goes beyond the parameters of a program, or
- want to access preceding chain processes in your program, or
- want to take errors when executing your program in modeling into consideration or you know multiple end states (see process type Decisions [Page 388])

we recommend that you implement your own process type.

Prerequisites

You need to have a good knowledge of the ABAP object-orientated language enhancement - ABAP Objects [External].

Process Flow

1. In the menu, choose *Tools* → *ABAP Workbench* → *Development* → *Class Builder*

   You can find additional information about the Class Builder in the SAP Web Application Server documentation under *BC - ABAP Workbench* → *ABAP Workbench: Tools* → Class Builder [External].

   The following `IF_RSPC_*` interfaces are available:

Interface IF_RSPC_*	**Description**	**Method is call**
EXECUTE	Executes the process. This interface is the main interface used to implement your own process type and is the only process chain maintenance interface that is obligatory.	Runtime
GET_INFO	Transfers the information for a process to the successor.	Maintenance a
GET_LOG	Returns the log for a variant and an instance to the process chain maintenance, either as a title and text or as a list of messages	Runtime and lo
MAINTAIN	Enables process variants to be created and maintained in the process chain maintenance. It also determines the header information for a variant so that it can be displayed in the process chain maintenance.	Maintenance

GET_VARIANT	Provides value help (F4) for variants. You need to implement this interface and/or interface IF_RSPC_MAINTAIN, in order to be able to add process variants to a chain.	Maintenance
GET_DEFAULT_CHAIN	Gives the default chain for a process. You can suggest, for each variant to be scheduled, which additional process it would make sense to schedule before or after the process.	Maintenance
CHECK	Consistency check at scheduling time for the process/chain You can carry out a check to determine: • the consistency of a process • the dependency on **other** processes in the **same** chain and • dependencies on **other** processes in **other** chains. • authorizations of current user to execute process	Maintenance
GET_STATUS	We recommend you implement this interface for distributed processes. Distributed processes, such as the loading process, are characterized as having different work processes involved in specific tasks. The interface allows you to request the status of the instance in the process chain maintenance(for distributed processes when follow-on events are not sent). Therefore, the system can also recognize terminations even for distributed processes.	Runtime and lo
CALL_MONITOR	Calls up the process monitor from the log view in the process chain maintenance.	Log view
TRANSPORT	Returns TLOGO entry.	Maintenance
CONTEXTMENU	Adds some functions to the context menu for a process in the process chain maintenance.	Maintenance
VARIABLES	Declares if the process can process runtime variables	Runtime
SIMULATION	Passes on and displays the simulation results	Runtime
MULTI_EVENT	Declares the possible end states of the process	Maintenance a

You can find further documentation for the IF_RSPC_* interfaces under *Tools → ABAP Workbench → Development → Class Builder* . For *Object Type,* enter the interface name and choose display. You can access the documentation in the subsequent screen using the *Jump* menu.

2. Enter the process type under *Settings → Maintain Process Types* (RSPROCESSTYPES view) in process chain maintenance..

   Then choose *Edit → New Entries*, allocate the technical name, short and long description for the process type, enter the name of your class and choose ABAP-OO Class as the *Object Type.* Make the other settings for your process type and save the entry.

3. If using the service class CL_RSPC_APPL_LOG, enter your process type as a sub-object to the log object BW_PROCESS.

Result

Your process is available in the process chain maintenance of the process category you assigned.

4.1.10.5 Process Chain Maintenance

Use

Process chain maintenance automates process flows in BI.

Prerequisites

If you want to include a load process in a process chain, you need to have already created an InfoPackage in the scheduler.

Features

You can execute various tasks in specified views, in the process chain maintenance. Different views, which you get by using the *Go to* menu, facilitate chain management. You define the process chain and schedule individual processes in the *Plan View*. You can use Drag&Drop to include the processes to be scheduled in the chain, beginning with the general start process. You can also then create links. Before you activate the chain, check for errors in the *Check View.* When you schedule the process chain, the chain run starts according to the start process selections. You can call the process monitor and display the logs for a process chain run in the *Log View*.

There are various ways of accessing the process chain maintenance.

- In the administration function area of the Data Warehousing Workbench, choose *Process Chains* from the navigation pane to call the process chain maintenance.
- In the modeling function area, you can call the process chain maintenance by choosing on the toolbar (or by choosing *Edit* → *Process Chains* from the menu).
- Choose *Process Chain Maintenance* on the toolbar to access process chain maintenance from the maintenance dialogs of the BI-supported processes.
  You then create a variant for the latest process, assign a description for the process chain, maintain additional associated process variants, and then create the links.

The display of the process chain plan view appears. Here you can either call an existing process chain, or create a new one. When you call the process chain maintenance from the maintenance dialog of a process, the system displays the process chain that contains these processes as a variant. If the process is used in more than one process chain you can select which process chain is to be displayed.

The chain is not locked in the display view of the process chain. Neither is the transport system called. If the active and edited versions of the chain concur, you can schedule the chain in the display view. You can access the *Change Mode* by choosing *Process Chain* → *Display/Change.*

Structure of the Process Chain Maintenance

In the left-hand area of the screen there is a navigation area that you can use to call navigation trees for process chains ( ) and process types ( ), as well as logs for process chains ( ). The navigation trees *InfoSources* and *Data Targets* simplify the insertion

of InfoPackages and processes for data targets. To facilitate the finding of required objects in these trees, you can execute a search using the icon.

The system displays the process chains (processes and their links) in a network graphic, in the right-hand area of the screen.

You can also display the chains in a tree display that is designed for screen reader devices. You can choose between the display variants by choosing *Settings → Network Display/Tree Display.*

You can call information about the status of the processes and links by choosing the *Legend* for a process chain maintenance view.

4.1.10.5.1 Plan and Check View Functions

Use

You can call the functions available in the plan and check views of process chain maintenance from the menu (or the toolbar in process chain maintenance), or in the context menu of the corresponding process. In addition, you can use Drag&Drop to transfer extra process chains to the plan view and call other process chains by double clicking on them.

Features

Functions in the Process Chain Maintenance Menu

Menu *Process Chain*

Menu Entry	Description
Create	You specify a technical name and a description for the process chain and a dialog box appears. In the dialog box, you create the start process for the process chain.
Copy	You use this function to copy the process chain you have selected, along with its references to process variants, and save it under a new name and description. If necessary, swap process variants with the variants needed in the new chain or delete or enhance the variants.
Save	You use this function to save the current status of the process chain.
Activate	You use this function to activate the process chain. You can only activate the chain if the system did not find errors during the check.
Reset to active version	You use this function to reset a process chain that has been changed and saved, back to its status when it was last activated.

Transport	You use this function to display the BI transport connection. You can transport the process chain and the dependent objects. **See also:** BI Transport Connection: Transporting Objects [External]
Attributes	You use this function to display or determine various attributes for process chains. **See also:** Display/Maintenance of Process Chain Attributes [Page 407]
Print	You use this function to print the network graphic for a process chain.
Save as image	You use this function to save the network graphic for a process chain as a jpg. file.

Menu *View*

Menu Entry	Description
Rotate	You use this function to switch between vertical and horizontal views.
Navigator on/off	You use this function to hide or display a movable navigator in the right-hand area of the process chain maintenance screen. It gives you an overview of the process chain that is being processed. It also allows you to enlarge or reduce the view by moving the mouse.
Enlarge	You use this function to enlarge your process chain view by ten percent.
Reduce	You use this function to reduce your process chain view by ten percent.
Adjust	You use this function to adjust the process chain view to the size of the right-hand screen area.

Detail view on/off	You use this function to display the detailed view for process chain maintenance. The process type, technical name of the variant, and description of the variant for a process are displayed here. Important elements of the worklist saved in the variant are also listed. For example, when updating the PSA, information about the data load process and data target is listed. A collection process is displayed several times in the detailed view. This multiple display helps you to visualize that a **job** is started for each **process** and for each **event** triggered by the process. You use the *Detail View Off* function to display the simplified view of process chain maintenance. Only the process types and variant descriptions for a process are displayed.
Remove from display	You use this function to remove a process chain from the display.

Menu *Execute*

Menu Entry	**Description**
Schedule	You use this function to schedule and execute the process chain in the background, in accordance with the settings in the start process.
Remove from scheduling	You use this function to delete the scheduling for the process chain. The chain is not executed.

Execute synchronously	You use this function to schedule and execute the process in the dialog, instead of in the background. The processes in the chain are processed serially using a dialog process. The system writes a log entry for each process. You can display the chain run in the log view. The separate processes themselves are processed serially, if this is possible. If this is not possible (depending on the process type), synchronous processing is terminated at this point and is only resumed when the other process has finished, and this has been communicated to the process chain. We recommend that you only execute simple chains with small datasets synchronously. In this case, the performance gain of executing in dialog (you do not have to schedule a background job) may outweigh the performance loss due to parallelization. Multiple runs can be processed in parallel, since there are no conflicts regarding the required background jobs.
Execute as simulated	You use this function to simulate the process chain. The system does not write log entries or process the data persistently. The process chain is displayed in the log view immediately after execution. If a process type supports this function, the context menu of the process contains the entry *Simulation Result*. This displays the result of the process.
Execute synchronous to debugging	You use this function to debug the execution of the process chain. The process chain is executed synchronously. If the chain contains separated processes, the system waits until these processes are completed.

Menu *Goto*

Menu Entry	**Description**
Plan view	You use this function to switch between the display and change modes for the process chain.

Check view	You use this function to switch to the plan view for the process chain. If the chain that you create does not correspond to the standard, the system displays a warning. You can ignore these warnings if you are sure that you are creating a chain that meets your requirements. Warnings do not affect how the chain operates at runtime. Your chain is checked in the check view and during activation. The system tells you if it discovers any errors.
Log view	You use this function to switch to the log view for the process chain. See also: Process chain log display
Job overview	You use this function to display a new mode in the job overview (transaction SM37).
Back	You use this function to undo the last navigation step. You use *Next Object* to repeat the navigation step.

Menu *Settings*

Menu Entry	Description
Maintain process types	You use this function to display process type maintenance. See also: Maintenance of Process Types [Page 415]
Maintain alert categories	You use this function to display alert category maintenance (table RSPC_ALERT_CAT). See also: Send Alerts for Process Chains [Page 411]

Default chains	You use this function to determine that the system does not propose processes for the process chain. If you insert a process into the chain that is linked to additional processes, the respective process variants are generated and inserted into the process chain automatically. These variants are suggestions and can be changed, replaced or removed from the chain as required. Variant maintenance is called when the change run performs automatic insert. If you want to insert a second process of the same type in the process chain and the selection conditions of the process overlap with those of the process already inserted in the chain, the system does not generate process variants. In this case, the system does not generate additional process variants, since it cannot guarantee that this is required. If you want to specify the processes that are to be included in a chain yourself, select *Do Not Suggest Processes*. As a result, the system does not suggest, generate, or automatically insert any processes into the chain. This setting is user-specific.
Tree display/network display	You use this function to switch between a tree display and a network-graphic display for the process chain. The tree display is designed for screen reader devices.

Displaying Additional Process Chains

In addition, you can display extra chains for the current process chain in the right-hand area of the screen in process chain maintenance. You can use Drag&Drop to move these from the process chain tree into the right-hand area of the screen.

If you link the processes of two chains vertically, the processes are added to a chain of another chain, and the superfluous chain is deleted.

If you display several chains, the system asks you which chain you want to select every time you perform an action. For example, if you have displayed several chains and want to remove one chain from the view, the system asks you to choose the chains that need to be removed when you execute the function.

If you save or activate, all the chains displayed are saved or activated respectively.

Displaying Other Process Chain

If you want to display another chain instead of the current process chain, select the one you want by double-clicking on the process chain tree.

Context Menu

If you select a process or a link by left-clicking on it, you can call up different functions by using the context menu (right-click mouse). The functions offered in the context menu depend

on whether you are in the plan, check, or log view, and whether process chains are already running. For more information about the respective functions, see Creating Process Chains [Page 402] and Process Chain Log Display [Page 412].

4.1.10.5.2 Creating Process Chains

Prerequisites

If you want to include a load process in the process chain, you need to have already created an InfoPackage.

You cannot load flat file data from a client workstation in the background. Therefore, you must store your data on an application server.

Creating Process Chains

You can either create a process chain directly in the process chain maintenance screen, or by using a maintenance dialog for a process:

Creating a Process Chain Directly in the Process Chain Maintenance Screen

In the BW Administrator Workbench:

1. Choose the *Process Chain Maintenance* icon from the AWB toolbar.

   The *Process Chain Selection* dialog box appears.

2. Choose *Create.*

3. Enter the technical name and a description of the chain, and confirm your entry.

   The *Add Start Process* dialog box appears.

4. Create a variant for a start process.

   a. On the *Maintain Start Process* screen, choose whether you want to schedule the chain directly or whether you want to start it using a metachain.

   b. If you choose to schedule the chain directly, enter the start date value for the chain under *Change Selections* and save your entries.

   The *Maintain Start Process* screen appears again.

   c. Save your entries, go back to the previous screen and confirm your entries in the *Add Start Process* dialog box.

   The *Plan View* of the process chain maintenance screen appears.

   In the left-hand area of the screen, a navigation area is displayed. In the right-hand area of the screen, the process chain is displayed.

5. Use Drag&Drop or double-click to add the relevant processes to your process chain.

   Choose *Process Types* to select the processes. This sorts the process types according to different categories. You can also call up InfoPackages and processes for the data target from the separate *InfoSources* and *Data Targets* navigation trees.

If you insert a process into the chain that is linked to additional processes by default, the respective process variants are generated and automatically inserted into the process chain. These variants are suggestions and can be changed, replaced or removed from the chain as required. Variant maintenance is called when the change run performs automatic insert.
You can turn this system response off using *Settings* → *Default Chains*.

For more information, see Plan and Check View Functions [Page 396].

Creating a Process Chain Using the Maintenance Dialog for a Process

In the maintenance dialog of a process that you want to include in a process chain:

1. Choose the *Process Chain Maintenance* pushbutton and create a process variant.
2. Save the variant and go back to the previous screen.

   A dialog box appears in which you enter a technical name and a description of the chain that you want to create.
3. Confirm your entries.

   The *Add Start Process* dialog box appears.
4. Create a variant for a start process.
   a. On the *Maintain Start Process* screen, choose whether you want to schedule the chain directly or whether you want to start it using a metachain.
   b. If you choose to schedule the chain directly, enter the start date value for the chain under *Change Selections* and save your entries.

   The *Maintain Start Process* screen appears again.
   c. Save your entries, go back to the previous screen and confirm your entries in the *Add Start Process* dialog box.

   The *Plan View* of the process chain maintenance screen appears.

   The various process categories, the application processes, and collection processes are displayed in the left-hand area of the screen. In the right-hand area of the screen, the process chain is displayed.

If the process that you used to create a chain is linked to additional processes by default, the respective process variants are generated and inserted into the process chain automatically. These variants are suggestions and can be changed, replaced or removed from the chain as required. Variant maintenance is called when the change run performs automatic insert.
You can turn this system response off using *Settings* → *Default Chains*.

For more information, see Plan and Check View Functions [Page 396].

5. Use Drag&Drop or a double-click to insert any additional relevant processes into your process chain.

   Choose *Process Types* to select the processes. This sorts the process types according to different categories. You can also call up InfoPackages and processes for the data target from the separate *InfoSources* and *Data Targets* navigation trees.

Additional Steps for Creating a Process Chain

6. When you add a process, you need to select a process variant or create a new variant. For collection processes, the system uniquely determines the variants.

Various functions for editing the process are available from the context menu:

Context Menu Entry for a Process	Function Information
Maintain variants	With the exception of the variants in the collection processes OR and EXOR, you can use this function to change all process variants.
Exchange variants	You can exchange the variants for an existing variant or a variant that is to be created.
Link with…	You can use this function to link a process to a successor process. You can choose from the processes that are not yet linked to the current process. They are grouped according to process type in submenus.
Display scheduled jobs	Once the process chain is active, you can use this function to display the jobs that have been scheduled.
Display all jobs	After at least one process chain run, you can display all of the scheduled jobs for a specific process and all of the jobs with which this process was run. The *Job Overview* appears and you can call the relevant job log.
Create message	You can also send messages to an application process of the chain, depending on the success or failure of the process. 1. To do this, using the context menu for a process, create another process variant of type *Send Message*. 2. If you maintain a message, first specify whether you want the message to be sent when the process has been completed successfully or unsuccessfully. Then choose *Next*. 3. A dialog box appears, in which you can select an existing process variant or create a new one. 4. If you create a new process variant, edit the document that is going to be sent and maintain a list of recipients. 5. Save your process variant and go back. The message process variant is now assigned to your application process. When the message is sent, the status information and the process log are also sent.

Debug loop	 Only use this function to debug a process run. Specify how long (in seconds) you want the delay to be between one event being triggered and the next process starting. You can capture the process in the debugger by using transaction SM37 (*Job Overview*) or SM50 (*Process Overview*).
Remove process	You use this function to remove a process from a process chain.
Manage data target	You use this function to call the data target administration for the following types of process variants: • Construct index • Delete index • Construct database statistics • Roll up filled aggregates • Compress InfoCube • Activate DataStore object data

7. Hold down the left mouse button to connect the processes with events.

   Before you do this, select the process underneath the process type row, and position the cursor over the required process. When you select the process type row, the whole process is moved into the plan view.

   From the context menu of a link, you can display the event or remove the link. To do this, select the link and right-click with the mouse.

8. If necessary, specify whether you want the event to be triggered after the previous process has been completed successfully or unsuccessfully, or whether you want the event to be triggered independently of the outcome of the process that precedes it. If the process that triggers the event has more than one option, choose the option after which the successor process is to be run (see process type Decisions [Page 388]).

9. Assign a display component to the process chain using *Attributes → Display Components.*

10. Maintain additional process chain attributes [Page 407] if necessary.

11. Check your process chain in the *Check View* and make any necessary corrections.

    The *Legend* explains the meaning of the different colors used to display the processes and links.

    From the context menu for a process, you can display the messages resulting from the check.

    During the check, the system calculates the number of parallel processes according to the structure of the chain (subchains are recursively taken into account here). The

result is compared with the number of background processes on the chosen server (or the total of all available servers if no server is specified in the attributes of the process chain). If the number of parallel processes is greater than the number of available background processes, the system highlights every level of the process chain where the number of processes is too high. The system produces a warning for these levels.

12. Save your process chain if it does not contain any errors.

Result

You can activate and schedule your process chain. After scheduling, the chain starts in accordance with the start process selections. For example, if you scheduled the start process directly and chose *Immediately* as the start date value, the chain run starts immediately after scheduling. In the *Log View,* you can display the reports for the chain runs.

See also:

Process Chain Display Logs [Page 412]

4.1.10.5.2.1 Variant Maintenance Tips

For various process types, variant maintenance in process chain maintenance is performed using the following table:

Object	Object Type	Object Name	Object Description
Symbol of object/process type is displayed	Description of object/process type is displayed	Technical name of object/process is displayed	Description of object/process is displayed

The following is valid when using this table:

- If you select an object type in the *Object Type* field, for example, InfoCube, then the execution of this process refers to the object specified in the *Object Name* field.
- If you select a process type in the *Object Type* field, for example *Execute InfoPackage*, then the execution of the process refers to the objects in the referenced process that was selected in the *Object Name* field.

  In the following cases, if you choose the process type *Execute InfoPackage* as the object type, the process is also executed based on the request that was loaded with this InfoPackage:

  - Activating ODS object data: The activation of data refers to the request in the selected InfoPackage.
  - Compressing an InfoCube: If you do not make any additional settings, compression is performed up to the last request loaded with the selected InfoPackage.
  - Rolling up filled aggregates: If you do not make any additional settings, roll up is performed up to the last request loaded with the selected InfoPackage.

You are also able to make process-specific settings in variant maintenance.

4.1.10.5.3 isplay/Maintenance of Process Chain Attributes

Use

You can display technical attributes, display or create documents for a process chain, and specify how process chains are to react at execution.

Features

You can display or maintain the following attributes for a process chain:

Process Chain → Attribute → ...	**Information**
Description (*Rename*)	You can change the name of the process chain
Display Components ( )	Display components act as sort criterion in process chain maintenance. Assigning process chains to display components makes it easier to access the chain you want. To create a new display component, choose the input help in the *Assign Display Components* window. In the *Display Grouping* dialog box that appears, assign a technical name and description to the display component.
Documents ( )	You can create and display documents for a process chain. For more information, see Documents [External].
Last Changed By ( )	Displays the technical attributes of the process chain: • When and by whom it was last changed • When and by whom it was last activated • Object catalog entry

Evaluation of Process Status	If you set this indicator and schedule a successor process *Upon Error* or *Always*, the system classifies all the incorrect processes of the chain as successful with regard to the overall status of the run. This indicator is relevant if you are using meta chains: You can classify errors in the processes of the subchains as 'unimportant' for the meta-chain run. The subchain is classified as successful, despite errors in processes of the subchain. If the successor of the subchain in the meta chain is scheduled *Upon Success*, the meta chain runs despite errors in the 'unimportant' processes of the subchain. This indicator does not apply to mailing or alerting; these functions are still triggered if there are incorrect processes with a successor processor scheduled *Upon Error*.
Polling Indicator	This indicator allows you to control how the main process responds if you have distributed processes. Distributed processes, such as the loading process, are characterized as having different work processes involved in specific tasks. You use the polling indicator to specify whether you want to keep the main process until the actual process has ended. Selecting the indicator: – Ensures a high level of certainty in the process flow – Informs external scheduling tools of the status of the distributed processes However, system resource consumption is higher and a background process is required.
Monitoring	With the indicator in the *Exclude Chain from Automatic Monitoring* dialog box, you can specify that you want to exclude a process chain from automatic monitoring using CCMS. By default, CCMS switches on automatic process chain monitoring. For more information about the CCMS context **Process Chains**, see BW Monitor in CCMS [Page 410].

Alerting	You use alert management to send alerts if errors occur during a process chain run. For more information, see Send Alerts for Process Chains [Page 411].
Background Server	Here you can specify the server or server group on which you want to schedule all the jobs in the chain. If you do not specify a server or server group, background management distributes the jobs across the available servers.
Editing Client	If you are using process chains in a client-dependent application, you specify the client in which you want the chain to be used here. You can only display, edit, schedule and execute the process chain in this client. If you do not maintain this attribute, you can display, edit, schedule and execute the process chain in all clients. *General Services* process variants that are included in a process chain that has this attribute are only displayed in the specified client. This attribute is transported. You can change it during the import by specifying an import client. In the target system, you have to create a destination for import post processing in the client that is specified (transaction RSTPRFC). The chain is then activated in this client after the import and scheduled if necessary. More information: Importing Client-Dependent Objects [External].
Execution User	In the standard setting, a BI background user executes the process chain (BWREMOTE). You can change the default setting so that you can see the user that executes the process chain and thereby the processes: You see this information in the *Job Overview* (transaction SM37). You can select the current dialog user who schedules the process chain job, or specify a different user. The setting is transported. The BI background user has all the required authorizations to execute all BI process types. Other users have to assign themselves these authorizations so that authorization errors do not occur during processing.
Job Priority	You use this attribute to set the job priority for all the jobs in a process chain.

4.1.10.5.3.1 BW Monitor in CCMS

Use

The Computing Center Management System (CCMS) has an alert monitor that supports you in monitoring and operating your SAP systems. The CCMS monitor sets of the alert monitor, together with the BW monitor, contain a selection of BI-relevant standard SAP monitoring trees and monitoring trees for process chains and consistency checks in the analysis and repair environment.

Activities

1. You get to the alert monitor from the menu by choosing *Tools → CCMS → Control Data/Monitoring → CCMS Monitor Sets* or by using transaction RZ20.

   In the overview tree of the CCMS monitor sets under the *SAP BW Monitor* node, you find the *BW Monitor*.

2. Select the *BW Monitor* and choose *Monitor → Start Monitor.*

   The monitoring trees for which the monitor objects are displayed are under the node for your BI system. Thus, a monitoring tree combines a group of logically related monitor objects.

   Monitor objects are specific components from your SAP system or its environment that you need to monitor.

   The monitor attributes are displayed for a monitor object. These describe data types that can be reported for a specific monitor object. Alerts can be deleted based on values that are reported for an attribute.

   For more information on the user of the Alert Monitor for CCMS, see Alert Monitor [External].

Monitoring Process Chains in CCMS

Monitoring Tree

The **Process Chains** monitoring tree contains all of the process chains that have run since the BI system was **last started**.

By setting the **transfer parameter for method execution** `DAYS_TO_KEEP_LOGS` in the method definition `RSPC_CCMS_STARTUP`, you can specify the process chain runs that you want to display from those that ended **before the BI system was last started**. By default, this parameter is set at seven days. This means that the process chain runs from the last seven days are displayed. You get the method definition through transaction **RZ21 -> *Methods -> Definitions***.

By double-clicking on the **Process Chains** monitoring tree, you can jump to process chain maintenance.

Monitor Objects

In the **Process Chains** monitoring tree, process chains that have run since the last start of the BI system are displayed as monitor objects, as well as those that were defined in transaction **RZ21**.

The nodes show the technical name of the process chain.

By double-clicking on a process chain, you jump to the maintenance for that chain.

Attributes are displayed for each process chain. These are:

- Text attribute

  The text attribute specifies the description of the process chain.

- Monitor attribute

  All runs of the chain (represented by the log ID as well as month, day, and time) for the last *n* days are displayed as monitor objects.

  You define the parameter *n* as the **transfer parameter for method execution** `DAYS_TO_KEEP_LOGS` in the method definition `RSPC_CCMS_AGENT`. If no changes have been made to the run for *n* days, it is removed from the display. However, you can still access the log directly from process chain maintenance. By setting the parameter to the value **`0FF`** you can generally exclude the process chains from being monitored by the CCMS.

  This agent runs a maximum of every ten minutes and monitors all of the runs displayed. The batch job for this is `SAP_CCMS_MONI_BATCH_DP` and is scheduled to run once an hour by default when you save the method. You can change the scheduling in transaction SM37.

  When you select a run and choose ***Display Details***, all messages for this run are displayed, as well as a message for every process that has finished. The default setting is to display the messages from the last 30 minutes. You can change this setting.

By double-clicking on a process chain run, you get to the log view of the process chain maintenance for this run.

Consistency Check Messages in the Analysis and Repair Environment (RSRV)

In the RSRV Checks monitoring tree, you can display the messages for the consistency checks that you executed in the analysis and repair environment (transaction RSRV).

4.1.10.5.3.2 Send Alerts for Process Chains

Use

You can send alerts if errors occur while processing processes or in the background processing for a process chain. Process chain maintenance uses alert management for this purpose.

The alert is sent to the user who scheduled the process chain. You can assign additional recipients by assigning roles or specifying individual users in alert category maintenance (transaction ALRTCATDEF).

The recipient's display program displays the alerts. The display program can be the Universal Worklist (UWL) in the portal, the alert inbox (transaction ALRTINBOX) or an application that calls alerts using API. You can also send alerts by E-Mail, fax or SMS. This is dependent on the configuration.

Features

If you have set the indicator in attribute maintenance for the process chain, an alert is sent by default for each process that has errors.

Before the system can trigger and send alerts, alert categories have to be assigned to the process types. The recipient, texts, priorities, escalation options, and so on can be determined in an alert category. The system determines the alert category as follows:

- Errors in background processing

  Alert category BWAC_PROCESS_CHAIN_FRAMEWORK is used for errors in background processing. For this category, fixed alert texts are delivered. These can be sent to the recipients if an alert occurs.

- Errors in the process

  The system searches table RSPC_ALERT_CAT for an entry for the process type (process chain maintenance → *Settings* → *Maintain Alert Categories*). You can change the entries in this table. It is locked against changes through SAP deliveries. This means that additional entries that are delivered can be transferred, but the existing entries are not overwritten.

  Maintain the table in two ways:

  - Assign a customer-defined alert category to a process type
  - Assign a class, which implements interface IF_RSPC_EVENT_RESOLUTION. Method if_rspc_event_resolution~get_category receives process variants and instances and determines the alert category dynamically.

    Since this class does not have to be identical to the class that implements the other process chain interfaces for this type, you can also implement alert triggers for SAP process types, without modification.

    You can include instances of alerting where the exception DONT_RAISE is triggered.

  If you do not enter a customer-specific alert category in the table, the system uses the alert category delivered for this process type.

  You can exclude a process type from alerting by selecting *No Alert* for the process type in table RCPC_ALERT_CAT.

  If table RSPC_ALERT_CAT does not contain entries for a process type, the system uses category BWAC_PROCESS_CHAIN_ERROR. This category uses the generic information on the process for the dynamic texts. This is also used when the system sends messages. Unlike sending messages using process chain maintenance, you can create an individual text for each process.

See also:

Alert Management [External]

Defining Alert Categories [External]

4.1.10.5.4 Process Chain Logs Display

Use

You can check process chain runs in the log view of the process chain maintenance.

Features

Calling the Log View of a Process Chain Run

You access the log view of a process chain either by choosing the *Log View* icon from the toolbar of the process chain maintenance or the *Logs* icon from the toolbar of the navigation area.

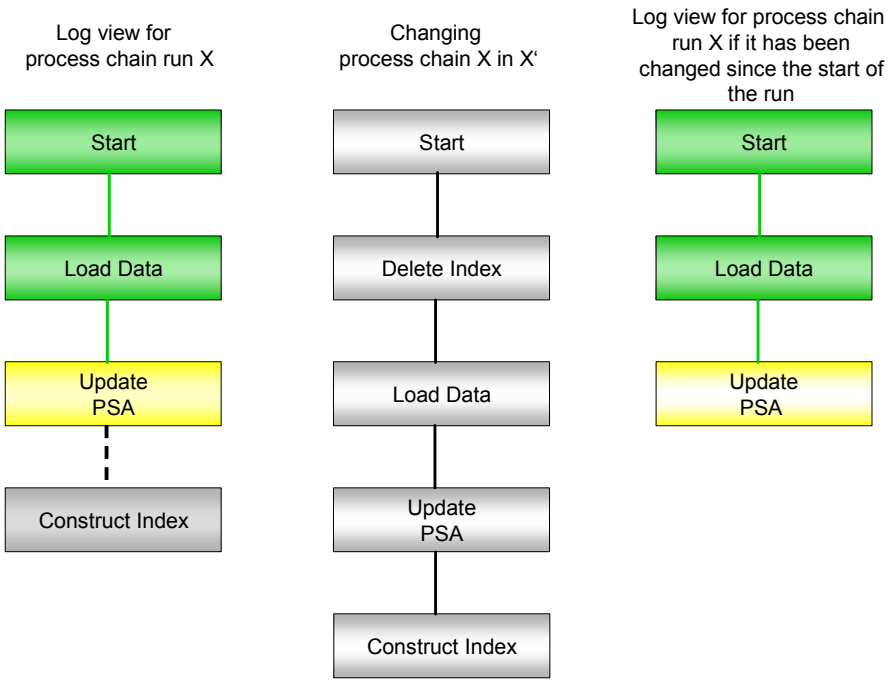
When you go to the log view, first choose the time frame for which you want to display the chain runs.

In the left-hand area of the screen, information about the time of creation, change, or activation as well as about the chain runs is displayed. Symbols display the status of the runs: Yellow indicates that the chain is active, green that the chain ended successfully, red that the chain ended with errors or was terminated. Unknown is displayed if the status is unknown, for example after an upgrade. Choose *Go to* → *Other Log* (on the process chain maintenance toolbar) to refresh the status display of the runs.

Double-click on the appropriate row to choose the log view for a run. You can refresh the log view for a selected run using the menu *View*.

You use the *Legend* to get information regarding the status of the processes and the links.

Depending on whether the chain has been changed since the last run, you can display processes that have not yet been run in the log view for a process chain. If the chain has not changed since the run to be checked, then the processes that have not been run are displayed in gray in the log view for this run. Also, the link for such processes is marked with dashes if the event has not yet been triggered. However, if the chain has been changed since the run to be checked, then the processes that have not yet been run and the events that have not yet been triggered are not displayed in the log view for this run.



Merging an Active Version

If the chain has been changed since the run to be checked, you can display the processes that have not yet been run in grey by choosing *View* → *Active Version.* This is particularly useful if the chain is to be continued after an error even if it has since been reactivated and/or scheduled.

Display Messages for a Process

By using *Display Messages* in the context menu for a process, you can call up the log. The logs are displayed in the dialog box that appears on the tab pages *Chain*, *Batch*, and *Process*.

- The tab page *Chain* contains information about the start and end of the process and the created instance.
- On the *Batch* tab page the logs for the job in which the process itself has run are displayed in the SAP List Viewer Grid Control. You access the job overview for your job using the *Batch Monitor* pushbutton.
- The *Process* tab page contains the process-based messages.

  This tab page is displayed if the process type writes its own log, or if the interfaces `IF_RSPC_GET_LOG` and/or `IF_RSPC_CALL_MONITOR` are implemented for the process type.

  You can use *Process Monitor* to get to this monitor with processes (the load process, for example) that have a special monitor attached.

Delete Process Chain Run Logs

If you want to delete the logs for a process chain and its assigned processes, choose *Log* → *Delete*. You select the currently displayed log on the next screen. You can also specify the time period you want to delete the log in.

Choose *Execute*. The system deletes all background jobs as well as the header and detail logs of the process chain framework.

If you set the indicator *Ignore Error*, the system proceeds with the deletion process despite any errors. If you do not set the indicator, the system terminates the deletion process.

You receive a list of deleted logs upon completion of the deletion process. The deleted run is no longer displayed in the log view and it cannot be restored.

Reselect the Process Chain Log

You can reselect the log for this process chain by choosing *Go to* → *Other Log* (on the toolbar of the process chain maintenance). The system updates the overview of the process chain runs according to your time selection. The system also refreshes the status of the runs.

See also:

Process Chain Attributes [Page 407] → Section: *Monitoring*.

4.1.10.5.5 Repairing/Repeating Terminated Processes

Use

In the chain log view you can restart any chain process (instance) that was terminated; this in turn ends the chain run.

Depending on the process type, there are two possibilities:

- You can repair a process. In this case the terminated instance is executed again.

  This enables you to restart processes that cannot be repeated with a new instance because the data to be edited is attached to the instance, as for example with a data transfer process. A data transfer process, for example, cannot be repeated with a new request number because the data itself is attached to the request.

- You can repeat a process. In this case a new instance is created.

Prerequisites

You can restart a terminated process in the log view of process chain maintenance when this is possible for the process type. In the plan view of process chain maintenance you can set whether a process can be repaired or repeated when terminated, under *Settings* → *Maintain Process Types* (table RSPROCESSTYPES).

Do not change the settings for the SAP process types. Otherwise this could lead to inconsistencies in the data handled by the process.

Activities

In the log view of the affected process chain, choose *Repeat* or *Repair* from the context menu of the terminated process.

If the process cannot be repaired or repeated after termination, the corresponding entry is missing from the context menu in the log view of process chain maintenance. In this case, you are able to start the subsequent processes. A corresponding entry can be found in the context menu for these subsequent processes.

4.1.10.5.6 aintenance of Process Types

Use

In the plan and check views of process chain maintenance, you can display a table overview which shows how the process types are configured. You can also apply this configuration to process types you have implemented yourself. Choose *Settings* → *Maintain Process Types*.

Features

You specify the following information in process type maintenance:

Field	What You Need to Know
Process Type	Specifies the technical name of the process type.
Short Description	
Long Description	

Class/Interface	Specifies the class in which the interfaces that are required for this process type are implemented.
Type of Object	ABAP OO class
Possible Events	Possible events that can be triggered after the process run: • Process finished “successfully” or “with errors“ • Process finished, status unspecified • Process is always the last process in a chain • Process finished with complex status You use this setting to control the process run on the basis of a logic with multiple values. You can define flexible process paths on the basis of multi-value decisions for predecessor processes that end successfully. The process type has to implement interface `IF_RSPC_MULTI_EVENT`. If you want to link this process to a successor process, you can select from the valid end statuses for a process that finished successfully. **See also:** Process type Decisions [Page 388]
Repeat	If an instance is terminated, you can repeat the process. If you repeat a process, the system generates a new instance. **See also:** Repairing/Repeating Terminated Processes [Page 414]
Rep. (Repair)	If an instance is terminated, you can repair the process. This means that you can repeat the process with the same instance. **See also:** Repairing/Repeating Terminated Processes [Page 414]
+ (Symbol)	Symbol that represents the process type in process chain maintenance.
GUID	The variant name is determined internally and is not determined in variant maintenance.
No Mail	Process chain mail service is not being used because the process type has implemented its own mail handling.
Process Category	Group of process types in process chain maintenance.

No.	Number for the position within the process category at which the process type is displayed in process chain maintenance.
Documentation Type	Documentation type for connecting documentation to the process type.
Docu Object	Documentation object for connecting documentation to the process type.
BW Only	Restricts a process type so that it can only be used in the BI standard client. In this case, the process type is not shown in the selection list in process chain maintenance when you use non-BI clients. The system highlights the corresponding processes as having errors when it checks the process chain.

4.1.10.6 Authorizations for Process Chains

Use

You use authorization checks in process chain maintenance to lock the process chain, and the processes of the chain, against actions by unauthorized users.

- You control whether a user is allowed to perform specific activities.
- You control whether a user is allowed to schedule the processes in a chain.

  The authorization check for the processes in a chain runs when the system performs the check. This takes place upon scheduling or during synchronous execution. The check is performed in display mode. The check is performed for each user that schedules the chain; it is not performed for the user who executes the chain. The user who executes the chain is usually the BI background user. The BI background user automatically has the required authorizations for executing all BI process types. In attribute maintenance for the process chain, you can determine the user who is to execute the process chain.

  See also: Display/Maintenance of Process Chain Attributes [Page 407] → *Execution User*.

Features

For the administration processes that are bundled in a process chain, you require authorization for authorization object S_RS_ADMWB.

To work with process chains, you require authorization for authorization object S_RS_PC. You use this authorization object to determine whether process chains can be displayed, changed or executed, and whether logs can be deleted. You can use the name of the process chain as the basis for the restriction, or restrict authorizations to chains using the application components to which they are assigned.

4.1.11 Non-Cumulatives

Use

A non-cumulative is a non-aggregating key figure on the level of one or more objects that is always displayed in relation to time. Examples of non-cumulatives include headcount, account balance and material inventory.

How you model the folder for non-cumulatives in the BI system depends on your scenario. According to how often non-cumulatives change and the total number of objects for which you want to calculate non-cumulatives, you should choose one of the following two options:

Non-Cumulative Management with Non-Cumulative Key Figures:

If the non-cumulatives change infrequently, you should choose non-cumulative management with non-cumulative key figures.

If you use non-cumulative key figures, an absolute non-cumulative value (the marker) and all non-cumulative value changes are saved in the fact table of the InfoCube. In this way, the retention and volume of data in the data load process is optimized. A data record is then only loaded to the InfoCube if a non-cumulative changes because of a transaction. Non-cumulatives can then be evaluated at any time in queries, using non-cumulative key figures.

The fact table for the InfoCube with non-cumulative key figures looks something like this (simplified):

Period	Article	Delta
2003001	AAA	1400
2003001	AAA	100
2003002	AAA	-150
2003003	AAA	-50
2003004	AAA	400
2003006	AAA	-300
2003009	AAA	1400

The fact table contains the transaction data. The first record in the table is the initialization. This entry does not remain physically in the table, but it is available for rebuilding the non-cumulatives.

The last record in this case is the marker (the InfoCube was compressed after this request). The non-cumulative for the period 2003005 is calculated as follows: 1400 – (-300) = 1700. This calculation takes place during the runtime of the query. Non-cumulatives can be displayed with this for time periods for which transaction data has not been loaded.

The use of non-cumulative key figures is recommended when the amount of transaction data is 20 % less than the granularity of the InfoCube.

Advantages of This Solution:

- The fact table is kept smaller.
- The history remains.

Disadvantages of This Solution:

- You cannot use the InfoCube together with another InfoCube with non-cumulative key figures in a MultiProvider.

- More administrative effort is required (for example, the InfoCube has to be compressed more often to keep the marker current).
- The query runtime can be affected by the calculation of the non-cumulative.
- Deletion of transaction data for material that is no longer current is not possible because deletion cannot be restricted by time.

See also Modeling of Non-Cumulatives with Non-Cumulative Key Figures [Page 420].

Non-Cumulative Management with Normal Key Figures (Cumulative Key Figures):

If the non-cumulatives change frequently, you should choose non-cumulative management with normal key figures. That is, choose cumulative values. Absolute non-cumulatives are then retained in InfoCubes for all objects for particular key dates (for example, the end of the month). These absolute non-cumulatives can be calculated from a DataStore object that is provided with non-cumulative value changes on an ongoing basis.

In this case, non-cumulative calculation takes place at query runtime. The marker is refreshed as a result of compression within the administration of an InfoCube with non-cumulative key figures.

The fact table for the InfoCube with normal key figures looks something like this (simplified):

Period	Article	(Delta)	Inventory value
2003001	AAA	(100)	1500
2003002	AAA	(-150)	1350
2003003	AAA	(-50)	1300
2003004	AAA	(400)	1700
2003005	AAA	(-)	1700
2003006	AAA	(-300)	1400

The fact table contains the non-cumulative values, but no delta (it is here to make it easier to understand). The non-cumulative value for a specific period can be determined using a key figure with an exception aggregation over the period.

The values for the key figures are saved in the granularity of the InfoCube. If the amount of transaction data is almost the same as the number of the most granular InfoObjects (for example, week multiplied by material) then the use of normal key figures is recommended.

Advantages of This Solution:

- You can use the InfoCube without restriction in a MultiProvider.
- The query runtime is not affected by the calculation of the non-cumulative.
- The deletion of transaction data for material that is no longer current is possible and easy.

Disadvantages of This Solution:

- The fact table becomes larger than it is when using non-cumulative key figures.
- In order to properly update posting in the future and in the past, the data first has to be loaded into a DataStore object and then into the InfoCube.

"How to..." Paper for Inventory Management Scenarios in BI

SAP provides a document that deals extensively with the technical features of non-cumulative management in BI.

To display this document in the SAP Service Marketplace, at http://service.sap.com/bi, choose *BI InfoIndex* from the right navigation area and then follow the *Inventory Management* link in the list. The document is called *How to...handle inventory management scenarios in SAP BW 2.x/3.x (pdf).*

4.1.11.1 Modeling Non-Cumulatives with Non-Cumulative Key Figures

Use

A non-cumulative key figure is modeled in BI using the corresponding field for the non-cumulative value change or the corresponding fields for inflows or outflows. You can determine the current non-cumulative or the non-cumulative at a particular point in time. You can do this from the current, end non-cumulative and the non-cumulative changes and/or the inflows and outflows.

The target of the special folder for non-cumulative key figures is to optimize the data transport into the BI system, to retain data, and to access the database when evaluating for reports in the BI system.

Whether the modeling of non-cumulatives with non-cumulative key figures is useful depends on your scenario. It is recommended that you use non-cumulative key figures for areas in which non-cumulatives do not regularly change completely, for example with warehouse stock (retail) or with the number of employees. For more information, see the extensive comparison in Non-Cumulatives [Page 418].

Structure

Model of Non-Cumulative Key Figures

A non-cumulative InfoCube is modeled with at least one non-cumulative key figure. These non-cumulative key figures are mapped using one key figure for non-cumulative changes or two key figures for inflows and outflows. Which option you choose depends on how you want to evaluate the non-cumulative key figure.

The key figures for non-cumulative value change or for inflows and outflows are normal cumulative key figures that have ‘summation’ both as aggregation and exception aggregation. Non-cumulative key figures always have summation as standard aggregation (aggregational behavior on the database, for example, upon compression or roll-up of aggregates). However, with reference to a time characteristic, they have an exception aggregation (in reporting) that is not equal to summation, as it would not make sense to cumulate non-cumulatives by time.

Non-cumulative key figures such as *Number of Employees* are cumulated using characteristics such as *Cost Center*. However, it is not meaningful to total the number of employees using different periods. With periods, for example, you can form an average.

With the cumulative value *Sales Revenue*, for example, it makes sense to cumulate the individual sales revenues using different periods, and using characteristics such as *Products* and *Customers*.

Example of the difference between non-cumulative and cumulative key figures:

Sales volume (cumulative value):

Sales volume 01.20 + sales volume 01.21 + sales volume 01.23 gives the total sales volume for these three days.

Warehouse stock (non-cumulative key figure):

Stock 01.20 + stock 01.21 + stock 01.23 does **not** give the total stock for these three days.

Technically, non-cumulatives are stored using a **marker** for the current time (current non-cumulative) and the storage of non-cumulative changes, or inflows and outflows. The current, valid end non-cumulative (to 12.31.9999) is stored in the marker. You can determine the current non-cumulative or the non-cumulative at a particular point in time. You can do this from the current, end non-cumulative and the non-cumulative changes and/or the inflows and outflows.

Queries for the current non-cumulative can be answered very quickly, since the current non-cumulative is created as a directly accessible value. There is only one marker for each combination of characteristic values that is always updated when the non-cumulative InfoCube (InfoCube that includes the non-cumulative key figures) is compressed. So that access to queries is as quick as possible, compress the non-cumulative InfoCubes regularly. (See Compressing InfoCubes [Page 735]) to keep the marker as current as possible.

For example, in month 03 the marker is read with three non-cumulative changes for a query. In month 04, the marker is updated so that the current marker has to be read with only one non-cumulative change for a query in month five. If the marker is not updated, it still has four non-cumulative changes to read

Data Transfer or Storage, and Aggregation for Non-Cumulative Key Figures

To optimize the data transport and data retention for non-cumulative key figures in the BI system, non-cumulative key figures are treated differently from cumulative values in both technical data transfer and storage:

- Non-cumulative key figures are mapped using one key figure for non-cumulative changes or two key figures for inflows and outflows.

  See also: Non-Cumulative Key Figures [Page 422]

- A non-cumulative InfoCube has to contain a time-reference characteristic, that means there must be a time-reference characteristic for exception aggregation of the non-cumulative key figure.

  See also: Time Reference Characteristics [Page 422]

- A non-cumulative key figure always has a time-related exception aggregation.

  See also: Aggregational Behavior of Non-Cumulative Key Figures [Page 423]

- In specific cases it may be necessary to determine the validity of a non-cumulative.

  See also: Validity Area [Page 426].

- Non-cumulatives are transferred in an initialization run and in the change runs that follow (initialization can also be omitted here).

See also Transferring Non-Cumulative Data into BW [Page 430]

Integration

In query definition and navigation, there is no difference in the way cumulative and non-cumulative key figures are dealt. Cumulative and non-cumulative key figures can be analyzed at the same time in a query.

You should not use more than one non-cumulative InfoCube (InfoCube with at least one non-cumulative key figure) in MultiProviders, because this could lead to incorrect query results.

The special treatment of non-cumulative key figures is only possible in the InfoCube. DataStore objects do not support this.

4.1.11.1.1 on-Cumulative Key Figures

There are two different ways to define non-cumulative key figures:

- **Non-cumulative key figure with non-cumulative changes:**

  Before you can define the non-cumulative key figure, an additional cumulative key figure containing the non-cumulative change must exist as an InfoObject.

- **Non-cumulative key figure with inflows and outflows**

  There has to be two additional cumulative key figures as InfoObjects for non-cumulative key figures - one for inflows and one for outflows. The cumulative key figures have to have the same technical properties as the non-cumulative key figure, and the aggregation and exception aggregation have to be SUM.

You can evaluate separately the non-cumulative changes on their own, or also the inflow and outflow, according to the type of chosen non-cumulative key figure in addition to the non-cumulative.

For aggregation, see Aggregational Behavior of Non-Cumulative Key Figures [Page 423]

4.1.11.1.2 ime Reference Characteristic

Use

If the InfoCube contains a non-cumulative key figure, then a time-based reference characteristic for the exception aggregation of the non-cumulative key figure must exist. There can be several time characteristics per InfoCube, but only one time reference characteristic. This means, **that the time-based reference characteristic is the same for all the non-cumulative key figures of an InfoCube**.

Integration

The time reference characteristic for an InfoCube when there are several time characteristics in the InfoCube is always the “most refined”, since all other times in the InfoCube are derived from this.

An InfoCube contains warehouse key figures that should be evaluated for the calendar month and calendar year. In this case, the calendar month is the most refined common time reference characteristic.

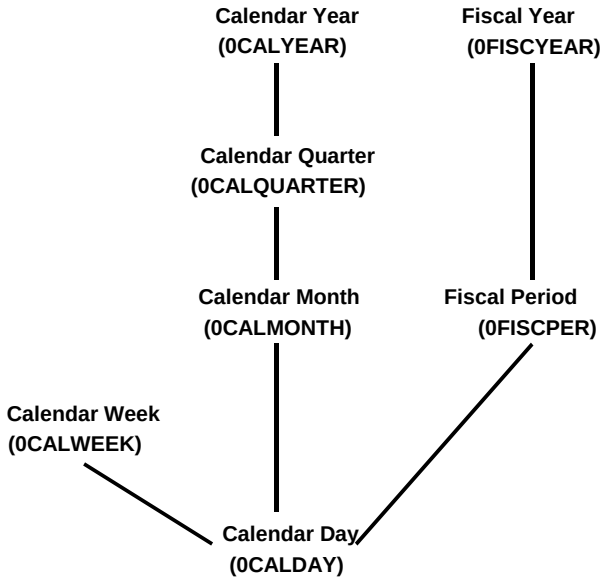
You can only maintain the time-reference characteristic and the fiscal year variant when updating an InfoCube with non-cumulative key figures. All other time characteristics are automatically derived from the time-reference characteristic. Therefore, the time-reference characteristic must not be left blank.

There is a difference between complete and incomplete time characteristics:

The complete time characteristics are the **SAP time characteristics** calendar day (0CALDAY), calendar week (0CALWEEK), calendar month (0CALMONTH), calendar quarter (0CALQUARTER), calendar year (0CALYEAR), fiscal year (0FISCYEAR) and fiscal period (0FISCPER). They are clearly assigned to a point in time. Only these SAP time characteristics can be used as time reference characteristics, since you must be able to derive time characteristics automatically from the most detailed time characteristic must be possible with the non-cumulative folder.

Incomplete time characteristics, such as 0CALMONTH2, 0CALQUART1, 0HALFYEAR1, 0WEEKDAY1 or 0FISCPER3 can be used in a non-cumulative InfoCube but cannot be a time reference characteristic, since they are not assigned to a specific point in time.

The following graphic gives an overview of the **hierarchy for SAP time characteristics**:



If you have a non-cumulative for a week and for a month in the same InfoCube at the same time, the roughest common time characteristic is calendar day. The time characteristic calendar day must be included in the InfoCube, so that it can function as a reference characteristic for time-based aggregation.

4.1.11.1.3 Aggregation Behavior of Non- Cumulative Key Figures

The aggregational behavior determines whether, and how, key figure values are aggregated in reports using the different characteristics or characteristic values. The aggregational behavior depends on whether you are aggregating using time characteristics or other characteristics.

You can see how aggregation affects the query result under Interpreting Query Results [Page 433].

If you add up all the cumulative values such as *Sales Revenue* using all characteristics (that is, time characteristics as well), the non-cumulative key figure relating to the time characteristic, the first value (FIRST aggregation), the last value (LAST aggregation) or the average is often taken.

You can find additional information about aggregate types under Tab Page: Aggregation [Page 108].

In the following example, the difference between the FIRST aggregation and the LAST aggregation is made clear. If one considers, for example, the aggregated values for 02.02.02, then the non-cumulative is considered 90 with the FIRST aggregation, which is the non-cumulative without receipts. The non-cumulative with the LAST aggregation is considered 110, which is the non-cumulative from 90 plus the receipts of 20.

Fact Table:

Date	Month	Material	Receipt
29.01.02	Jan	4711	30
30.01.02	Jan	4711	10
31.01.02	Jan	4711	20
01.02.02	Feb	4711	30
02.02.02	Feb	4711	20

Query:

Date	Non-Cumu (LAST)	Non-Cumu (FIRST)
28.01.02	0	0
29.01.02	30	0
30.01.02	40	30
31.01.02	60	40
01.02.02	90	60
02.02.02	110	90
03.02.02	110	110

There are two possible kinds of aggregational behavior for non-cumulative key figures:

- The **standard aggregation** specifies how a key figure is compressed using all characteristics (but not time characteristics).
- The **exception aggregation** specifies how a key figure is compressed using all time characteristics.

Exception aggregations with regard to time

Every key figure has a standard aggregation and an exception aggregation. Non-cumulative key figures always have summation as standard aggregation, whereas time characteristics have an exception aggregation of not equal to summation.

The non-cumulative key figure *Warehouse Stock* is aggregated using *Summation* for characteristics that are not time-related such as *Articles* or *Stock*.

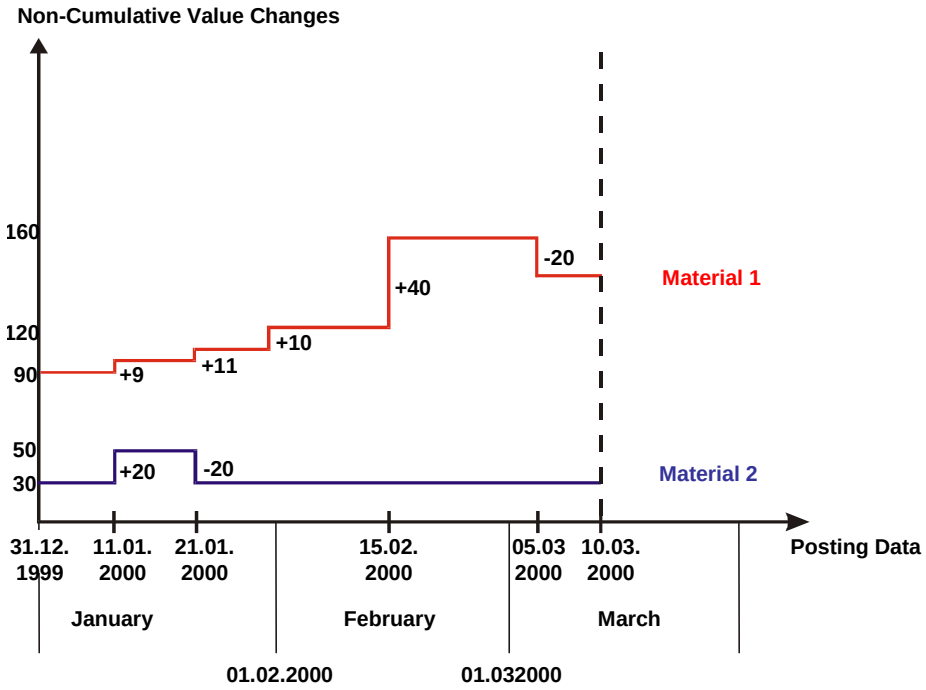
For time characteristics such as *Calendar Month*, however, the non-cumulative key figure *Warehouse Stock* has the exception aggregation *Last Value*.

Meaningful aggregations for non-cumulative key figures are primarily *Average Weighted According to Calendar Days* (AV1) and *Last Value* (LAS). Additional, possible exception aggregations for non-cumulative key figures are listed in the following table.

Exception aggregation for non-cumulative key figures

Technical name	Description
AV1	Average (weighted with the number of calendar days)
AV2	Average (weighted with the number of working days according to the factory calendar with the ID 01)
FIR	First value
LAS	Last value
MAX	Maximum
MIN	Minimum

The time at which non-cumulatives were posted for different materials is displayed in the following graphic. The evaluation results for the non-cumulative for *Material 1*, for exception aggregation *Average*, and the exception aggregation *Last Value*, are listed in the following tables, where they are displayed once by calendar month and once by calendar day.



Drilldown on the non-cumulative value for Material 1 by calendar month

	Average (calendar day)	Last value

January	100	110
February	140	160
March	150	140

Drilldown on the non-cumulative value for material 1 by calendar day

Note that non-cumulatives that are evaluated by calendar day, both for the average and for the last value, always produce the same result. The reason for this is that *Calendar Day* is the smallest unit of time to which the data is transferred. This always occurs when you drilldown on the most detailed time characteristic.

	Average (calendar day)	Last value
01.01.2000	90	90
02.01.2000	90	90
03.01.2000	90	90
04.01.2000	90	90
...	...	...
09.01.2000	90	90
10.01.2000	90	90
11.01.2000	99	99
...	...	...

4.1.11.1.4 Validity Area

Use

Only the currently valid end non-cumulative and the non-cumulative changes are stored in the database. To be able to evaluate the non-cumulative (for example, to calculate mean values or to execute a drilldown according to period), you need to know for which time interval you can calculate the non-cumulative. Otherwise, a non-cumulative from 01.01.0000 to 12.31.9999 is displayed. The minimum and maximum loaded values of the time reference characteristic (for every feature of the validity table) are thus used as standard as a threshold for the validity area.

Features

Validity area: The differing time-based validity of non-cumulatives is mapped using a validity area. The validity area describes in which time period non-cumulatives have been managed.

Normally this time interval is valid for all records for the InfoCube, for example for all cost centers, materials, etc. The validity interval is comprised of the minimum and the maximum of all postings.

If the first values are posted for a product group on 12.31.1999 and a non-cumulative change was posted for the last time on 3.10.2000, the validity area is the time interval from 12.31.1999 until 3.10.2000.

In certain cases it may be necessary, however, to manually adjust the validity, for example when a characteristic value is only valid for a restricted time period. In this case, you should define a validity area, especially to be able to guarantee correct calculation of averages.

Another example: If the data from various sources systems has been loaded to the InfoCube at different times, it can be sensible to keep the validity area for the respective partial areas.

Source system 1: Delivers values for the current month

Source system 2: Delivers values for the previous month

If the source systems deliver values at different times, aggregation can become affected over a period of time and contain inaccurate values. The calculated monthly average is too high, for example (see example at the end of this section).

All validity areas for a non-cumulative InfoCube are stored in a validity table. This validity table automatically contains the most detailed, selected time characteristic of the InfoCube, the time-reference characteristic.

Validity-determining characteristics: Characteristics that determine the temporal validity area of the non-cumulative in an InfoCube.

Validity-specific characteristics are, for example, characteristics that specify the assignment to a source system. See also: Non-Cumulative Parameter Maintenance [Page 136].

Aggregation using validity-determining characteristics:

With aggregations that do not calculate an average (MIN, MAX, FIRST, LAST), virtual entries are added in the validity interval margin so that all intervals have the same start and end points. The aggregation is then executed. These virtual entries are highlighted in a special way in the results view. Without these virtual entries, the totals row would deliver inaccurate results. This is not valid for the aggregation type AVERAGE.

In the following example, the receipts for plants A, B, and C are displayed. The virtual entries are indicated in a special way in the query. Looking at plant B, then there are only receipts in March and April. For January and February, the virtual entry 120 is set, because this is the non-cumulative excluding the receipts from March. For May the virtual entry 150 is set, which is the final non-cumulative for April. In the totals row, the total of virtual entries and the actual non-cumulatives is calculated.

Receipts:

```
          Jan   Feb   March   Apr   May       Marker
Factory A:      +30         +20   +40   +10       100
Factory B:                  +10   +20             150
Factory C:                        +20   +30        50
```

Query:

```
                                                LAST
Factory A:       30    30    50    90   100     | 100
Factory B:     [120][120] 130   150 [150]     | 150
Factory C:       [0]   [0]   [0]   20    50     |  50
  ----------------------------------------------
TOTAL :         150   170   180   260   300     | 300
```

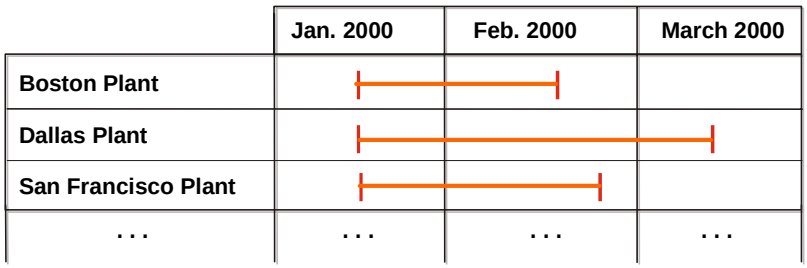
SAP recommends that when you create a non-cumulative InfoCube, you use validity-determining characteristics only in the specified cases. If you have too many validity-determining characteristics or a validity-determining characteristic with a lot of characteristic values, performance decreases considerably.

If you decide later that you require more validity-determining characteristics, you can modify the selection using the report RSDG_CUBE_VALT_MODIFY. In this report, the non-cumulative InfoCube is only changed to the extent that the new validity-determining characteristics are selected and the validity table is reconstructed. The structure of the non-cumulative InfoCube remains the same. You do not have to reload the transaction data for it.

For each combination of characteristic values for the validity-determining characteristics, the validity area is, by default, the interval between initialization (or the first change in non-cumulative) and the last posted non-cumulative change for this combination. That is, the validity area is created from the posting data from when the data was loaded.

When evaluating in reporting, the non-cumulative for the requested time period is evaluated using the current final non-cumulative and the corresponding non-cumulative changes (meaning that the non-cumulative is defined at every moment within the validity area). In this way, the non-cumulatives for time periods for which no change was posted are also identified.

Assuming that every plant delivers its data to the BI System separately at different times, the characteristic *Plant* has to be validity-determining. If you also assume that the characteristic values are *Boston Plant, Dallas Plant,* and *San Francisco Plant*, the validity intervals appear as follows:



For the Dallas plant and the San Francisco plant, the validity table is maintained as follows:

Plant	From (fixed)	To (fixed)
Dallas	001.2000	003.2000
San Francisco	001.2000	002.2000

The following data is then displayed in the query:

Plant	January	February	March	Result	Comments
Dallas: Material Stock	700	700	900	766,66	(∅ per month)
Dallas: Goods Receipts	200	0	200	400	(SUM)
San Francisco: Material Stock	200	300	(300)	250	(Not 266.66!)
San Francisco: Goods Receipts	50	100	(0)	150	(SUM)

In the query, the average amount from January and February is given as the result for the San Francisco plant since only these time periods have been defined as valid. If a validity area had not been defined, the average given would have been too high (266.66).

See also:

Maintaining the Validity Area [Page 429]

4.1.11.1.4.1 Maintaining the Validity Period

Prerequisites

The user-specific change to the validity table can be carried out only after the corresponding InfoCube has been activated with a Release status of 2.0B (or higher).

Procedure

The minimum and maximum loaded values of the time reference characteristic (for every feature of the validity table) are used as standard as a threshold for the validity period toolbar. You can change the validity of the individual characteristic values independently of the data loaded.

1. Call transaction RSDV

2. You get to the window *Report on Validity Slice Maintenance*. Specify the InfoCube here, for which you want to maintain the validity area, and select *Execute*.
3. In the window *Editing the Validity Slice for Non-cumulative InfoCubes,* select *Display/Change*.
4. You can now determine, for each individual feature, the update method for the lower ('from' mode) and upper ('to' mode) limits of the validity area. You have the following selection options:

- Feature " "

  This mode corresponds to the standard setting, meaning that for every feature, the minimum or maximum (time-related) loaded value is taken.

  The current, valid value is displayed in the *From (Fixed)* (for the start value) or *To (Fixed)* (for the end value). You cannot change these values.

- Feature "F"

  Here you can specify a fixed start or end date. Enter the values you want in the fields *from fixed* (for the start value) or *to fixed* (for the end date).

  The input format must be adjusted to the corresponding time characteristic.

  If 0CALDAY is the time-reference characteristic, then the input format has to correspond to the date format day, month, year as specified in the global setting, for example, 01.01.2000 or 2000/01/01. If 0CALMONTH is the time-reference characteristic, then the input format must contain month and year correspondingly.

- Feature "R"

  Here you can specify the interval thresholds relative to the current time. Enter, as whole numbers (positive or negative), the values you want in the field *from rel.* (for the start value) and *to rel.* (for the end value). The specified values describe movement in units with time reference characteristics with reference to the current time.

  If the number '+1' is entered in the field *rel. to,* for time-reference characteristic 0CALDAY, this means that the validity period threshold is the current next day. However, if 0CALMONTH is the time-reference characteristic, then the threshold for the validity period is the next current month.

5. Save your entries.

Result

The modified validity table is taken into account in reporting.

4.1.11.1.5 Transferring Non-Cumulative Data into the BI System

Features

Non-cumulatives can first be included through initialization or by loading a non-cumulative change. With **initialization** you are only able to post one initial non-cumulative for each

combination of characteristics in the InfoCube. **This causes non-cumulative changes**. There is one InfoSource for the initial non-cumulative and one for non-cumulative changes.

See also, Business Content objects under inventory management (*SAP Library → SAP-NetWeaver Components → Information Integration → Business Intelligence → BI Content → Supply Chain Management → Supply Chain Performance Management → Inventory Management*).

Data Flow with InfoSource 3.x:

InfoSource for initial non-cumulative: Transfers the current non-cumulatives once from the source system to the BI system. You need this for example if a system is connected to a BI system at a later date. By doing this you can ensure that the non-cumulative situation in the BI system is the same as the situation in the source system.

InfoSource for non-cumulative changes: Transfers current transactions, goods receipts, goods issues, and so on.

If an initial non-cumulative is to be loaded for a characteristic value combination, then this occurs in the **first** load procedure with the InfoSource for the initial non-cumulative. If a non-cumulative change is posted first, then no further initial non-cumulative can be posted for this characteristic value combination. The initial non-cumulative is then equal to zero. In the query, the first delta, which actually contains the initial value, appears to be especially high.

For other combinations of characteristics, initialization is permitted. If you load an initialization in an InfoCube that already has data, a warning appears. However, you can continue with the initialization. Both initializations are added to the query. However, you can no longer compress the request. Before compression, you have to delete the request with the second initialization from the InfoCube.

When loading the data, the validity period is determined and updated correspondingly, for every combination of validity-defining characteristic and per request.

An initial non-cumulative is firstly loaded for plant A. Non-cumulative changes are then posted for this plant. Subsequently, a non-cumulative change is posted in the InfoCube for plant B. Data for plant C is then posted in the InfoCube with an initial non-cumulative at first and then with non-cumulative changes.

Only non-cumulative changes, not initial non-cumulatives, can now be posted for all three plants.

4.1.11.1.6 Process Flow for Evaluating Non-Cumulative Key Figures

For evaluating non-cumulative key figures, you can usually use the corresponding **Business Content** See also: Installing Business Content [Page 435]

For information about which Business-Content objects are delivered, see inventory management (*SAP Library → SAP NetWeaver Components → Information Integration → Business Intelligence → BI Content → Supply Chain Management → Supply Chain Performance Management → Inventory Management*). In particular, see

- he material movements InfoCube(0IC_C03)
- The material stocks InfoSource (from 3.1 Content) (2LIS_03_BX), material movements InfoSource (2LIS_03_BF) and revaluations InfoSource (2LIS_03_UM), and their DataSources.

If you want to model your own objects, proceed as follows:

1. **Source system**

   There are two extraction modes for non-cumulatives in the source system. The DataSources for non-cumulatives have to support one of these extraction modes.

2. **Modeling:**

   Determine in which way you wish to evaluate the non-cumulative key figure. If you only want to evaluate the non-cumulative or the non-cumulative changes as well as the non-cumulative using a particular time period, choose modeling as a non-cumulative key figure with non-cumulative changes. If you also want to evaluate the inflows and outflows separately, choose modeling as a non-cumulative key figure with inflows and outflows.

3. **Key figure definition:**

   The non-cumulative change or inflows and outflows that you require for the definition of the non-cumulative key figure must already be created as InfoObjects when the non-cumulative key figure is created. Therefore, where necessary, define either the non-cumulative change or the inflow and outflow as normal cumulative key figures first. In type definition, these key figures must correspond to the non-cumulative key figures that still have to be defined. The non-cumulative changes or inflows and outflows have to have the aggregation and exception aggregation type ‘summation’.

   Define the non-cumulative key figure by assigning to it the previously defined non-cumulative change or in and outflows.

   Choose the exception aggregation you want to use. Standard aggregation is always “summation”.

4. **InfoCube definition:**

   If you transfer the non-cumulative value into the InfoCube, the respective cumulative values (non-cumulative change or in- and outflows) are automatically included in the definition of the InfoCube.

   In the definition of the InfoCube, include all time characteristics that you want to use to evaluate the InfoCube data.

   Additionally, in *Extras → Maintain Non-Cumulative Values*, you can choose one or more characteristics that further specify the validity of the non-cumulative. These characteristics (down to the most detailed time-reference characteristic) are also listed in the validity table of the non-cumulative.

   Refer to the SAP Notes for defining validity tables. It is not normally necessary to include additional validity-determining characteristics in the validity table.

   Activate the InfoCube.

5. **Loading data:**

   If you want to load an initial non-cumulative, load this **first** and then load the non-cumulative changes.

   If a non-cumulative change is updated when loading data for the first time, no further initial non-cumulative can be loaded for this characteristic value.

   When loading data, the validity areas for the individual validity-defining characteristic values are automatically updated.

6. **Query definition:**

In the InfoCube both the non-cumulative key figure and the non-cumulative change, or in- and outflows, can be selected for query definition.

With evaluations in reporting, there is no difference in the way cumulative and non-cumulative values are handled. Cumulative and non-cumulative key figures can be evaluated at the same time in a query, since the key figures are automatically aggregated correctly.

4.1.11.1.7 Interpreting Query Results

Before you can even think about interpreting the result of a query on a non-cumulative InfoCube, you need to know the sequence of the aggregations.

The OLAP processor executes the aggregations in the following sequence:

1. Normal aggregation
2. Exception aggregation with respect to the reference characteristic
3. Currency and units aggregation

Note: Even in cases where you haven't included the reference characteristic in the query, an aggregation is implicitly executed using the reference characteristic.

Example:

Your query has a key figure with the exception aggregation LAST and the reference characteristic 0CALDAY. You have loaded the following data into your InfoCube:

Plant	Posting Date	Value
1	15.06.	25
1	16.06.	15
2	15.06.	20

You now have the following drilldown by plant in your query:

Plant	Key Figure (SUM/LAST)
1	15 (last posted value related to plant 1 - 16.06.)
2	20 (last posted value related to plant 2 - 15.06.)
Result	15 (last posted value related to plant 1+2 - 16.06.)

This result comes from the fact that the normal aggregation is carried out with respect to the plants. The exception aggregation now has an influence in the construction of the result. As only one value was booked for plant 1 for the last date (16.06), only the value for plant 1 is included in the result, that is, 15.

4.1.11.1.8 Excurs.: Mod. Non-Cumul. Key Figs with Differing Time Ref. Chars

If an InfoCube contains a non-cumulative key figure, then a time-based reference characteristic for the exception aggregation of the non-cumulative key figure must exist. There can be several time characteristics per InfoCube, but only one time reference characteristic. This means, **that the time-based reference characteristic is the same for all the non-cumulative key figures of an InfoCube**.

Different Time Reference Characteristics

If you have characteristics for which you manage non-cumulatives and which refer to a data object in different stages of editing, such as delivery stock, order quantity and billing quantity, then these non-cumulative key figures all have differing time references. Therefore, these non-cumulative key figures cannot be evaluated like this in a joint InfoCube.

Modeling Proposal

You can map the different time references using a **characteristic transaction**.

You then have a key figure ‘non-cumulative’ with the most detailed time reference characteristic calendar day, and a characteristic ‘transaction’ with the characteristic values ‘delivery’, ‘order’, ‘billing’. This means you can store the non-cumulative for the differing transactions in one, single InfoCube.

For evaluation in reporting, you can then use the restricted key figure ‘non-cumulative’ that is restricted to one of the characteristic values of the ‘transaction’ characteristic, as a structure element in the query definition.

In this way, you can evaluate the ‘delivery stock’, the ‘order quantity’, and the ‘billing quantity’ using restricted key figures.

By doing this, you can minimize data transfer and storage, and reduce the number of key figures.

4.1.11.1.9 Performance Tips

To attain good performance for a query on non-cumulative InfoCubes, you should take note of the following:

Compression:

Compress all requests in the non-cumulative InfoCube, or at least most of them.

The performance of a query based on a non-cumulative InfoCube depends heavily on how the InfoCube is compressed. If you want to improve the performance of a query of this type, first check – in so far as this is possible - whether the data in the InfoCube should be compressed. You should always compress data when you are sure that the request affected will not need to be deleted from the InfoCube.

Validity Table

Use as few validity-determining characteristics as possible.

The number and cardinality of the validity-determining characteristics heavily influences performance. Therefore, you should only define characteristics as validity-determining characteristics when it is really necessary.

Time Restrictions in the Query

As far as possible, restrict queries based on non-cumulative InfoCubes to time characteristics.

The stricter the time-based restriction, the faster the query is generally executed, as the non-cumulative is reconstructed if the number of times is smaller.

Time Drilldown in the Query

If you no longer need the average, split a query on a non-cumulative InfoCube (which contains both key figures with LAST aggregation and key figures with AVERAGE aggregation) into two queries.

With non-cumulative key figures with the exception aggregation LAST, the time characteristic included in the drilldown makes a difference to performance. If, for example, both *Calendar Day* and *Calendar Month* are included in the InfoCube, drilldown by month is faster than drilldown by day, because the number of times for which a non-cumulative has to be calculated is smaller.

For the other types of exception aggregation (average, average weighted with factory calendar, minimum and maximum), this rule is not valid as in these cases, the data is always calculated on the level of the most detailed time characteristic first before exception aggregation is performed.

Totals Rows

Hide the totals row in the query when not required.

Depending on the type of aggregation being used, the calculation of totals rows can be very time-consuming.

4.1.12 Installing BI Content

Use

Before you can work with BI Content objects, you have to convert them from the SAP delivery version (D version) into the active version (A version). For more information, see BI Content (Versions) [External].

We recommend that you install BI Content objects in the following cases:

- After a BI Content release upgrade

  An upgrade to a new BI Content release usually contains a large number of new and modified BI Content objects. If you want to use the new and modified BI Content, you have to reinstall the BI Content objects. Refer to the Release Notes published with the upgrade. In some cases, we recommend explicitly that you install the A versions of certain objects.

- After installing a BI Content support package

  Refer to the SAP Notes that are delivered with each Content Support Package. These SAP Notes contain information on why certain BI Content objects are being redelivered.

Procedure

The following is an overview of the individual steps in the procedure: These steps are subsequently explained in more detail.

1. Navigate to the *Data Warehousing Workbench* to install BI Content (object selection, installation settings, starting installation).
2. Assign relevant source systems.
3. Group objects to be included, determine mode of collection for the objects.
4. Determine the view of the objects.
5. Transfer the objects to *Collected Objects*.
6. Check the settings for the *Collected Objects* with reference to the *Install*, *Match or Copy* and *Active Version Available* functions.
7. Make settings in the selection list and *Transfer*.
8. Correct errors.

1. Navigate to the *Data Warehousing Workbench* to Install BI Content

From the main Business Intelligence menu, choose *Modeling* → *Data Warehousing Workbench: BI Content.* If you are already in the Data Warehousing Workbench, select the *BI Content* functional area by clicking on the corresponding pushbutton in the left-hand navigation window, or by choosing the path *Goto* → *Install BI Content.*

The Data Warehousing Workbench for installing BI Content has three navigation windows:

- In the left-hand window you determine the view of the objects in the middle area of the screen.
- In the middle window, you select the objects that you want to activate.
- In the right-hand window, you make the settings for installing the BI Content. The right-hand window also contains an overview of the objects you have selected, and it is here that you start installation of BI Content.

In the Data Warehousing Workbench, you use the *Navigation Window On/Off* pushbutton in the toolbar to display or hide the left-hand navigation window. The rest of this section assumes that the left-hand navigation window is displayed.

2. Assign Relevant Source Systems

If you want to assign a source system, select the *Source System Assignment* function. The *Choose Source System by Default?* dialog box appears.

Select one or more source systems by setting the corresponding indicators in the *Default Assignment* column.

Only ever select the source systems that you really need, otherwise you may have to wait unnecessarily when objects are collected.

The assignment of the source system is only relevant for source-system dependent objects (such as transfer rules, file DataSources, and InfoPackages).

If more than one source system is available, only those objects assigned to the specified source system are collected ready for the transfer. Objects that have not been assigned to the specified source systems are ignored.

For more information about the special features inherent in activating process chains that can reference source-system dependent objects, see the Customer Content documentation, under Process Chains and Process Variants [Page 470].

If you do not select a source system, **all** the source systems are assigned automatically. You can change your selection later using the *Source System Assignment* function.

3. Group Objects To Be Included, Determine Mode of Collection for Objects

Make the settings you require from the following selection lists on the right-hand side of the screen:

- **Grouping**

  Choose the objects that you want the system to include. The groupings combine the objects from a particular area. You have the following options:

  - *Only Necessary Objects* (default setting)
  - *Data Flow Before*
  - *Data Flow Afterwards*
  - *Data Flow Before and Afterwards*

  For more information about groupings, see Collection Object Grouping [External].

  If you change the default setting (*Only Necessary Objects)*, the new setting becomes the default setting for your user.

  The grouping selection has an impact on system performance during BI Content installation. For more information, see View of Objects and Object-Specific Recommendations [External].

- **Collection Mode**

  Select how you want to collect the objects:

  - *Collect Automatically* (default setting): The data is collected when the objects are selected.
  - *Start Manual Collection:* The data is only collected when you choose *Collect Dependent Objects*.

    Set the collection mode to *Start Manual Collection*. You can select all the objects without having to wait.

4. Determine View of Objects

In the left-hand navigation window, specify how you want the objects to be displayed. For more information, see View of Objects and Object-Specific Recommendations [External].

5. Transfer the Objects to *Collected Objects*

In the central area of the screen, select the objects that you want to install, and drag and drop them into the right-hand *Collected Objects* area of the screen.

The *Find Object* function allows you to use strings of characters (for example, 0CUST) or wild card searches (for example, 0CUST_*B) to search for objects.

Input help is available for every type of object: Double-click on the *Select Objects* icon in the tree structure of the corresponding object type to display the *Input Help for Metadata* screen. Select the required objects. Choose *Transfer Selection.*

If you implement BI Service API Releases lower than 7.0 in the source system, you have to install the active version of the BI Content DataSources in the source system and replicate them in the BI system before you can use them for transferring data into BI. For more information, see Installing BI Content DataSources [Page 540] and Metadata Upload for SAP Systems [External].

In the *Collected Objects* area of the screen, the system displays the selected objects and all dependent objects. Collected objects are stored by default in the repository cache [External]. This reduces the time it takes to access these objects when you want to use them again.

When you transfer objects into the *Collected Objects* area of the screen, these objects are also added to the tree structure of the corresponding object type in the central area of the screen and stored for your user. This **personal object list** can be called up each time the program is restarted.

If you want to remove objects from your personal list, select the objects that you want to remove and choose the *Remove Object from Display* option from the context menu or click on the icon.

Objects that are listed in several tree structures can only be changed in the place where they first appear. All additional instances of these objects are grayed out so you cannot modify them.

6. Check Settings for *Collected Objects*

Check the following columns in the *Collected Objects* area of the screen:

- ***Install***

  The following BI Content objects are highlighted in this column by default:

  - Objects that are being transferred for the first time. There is not an active version of these objects in the system.
  - BI Content objects that have been redelivered in a new version. These objects can be identified by the Content time stamp in the corresponding object tables.

  When setting this indicator, check whether the checkbox refers to a folder of an individual object: If the checkbox refers to a folder, the indicator is set for all the objects that belong to this folder. If the checkbox refers to an individual object, the indicator is set for a single object and the indicators for the other objects in the folder are not changed. The same applies if you deselect this indicator.

  In the context menu, the following options are available for the installation:

a. *Install All Below*

The object in the selected hierarchy level and all objects in the lower levels of the hierarchy are selected as to *Install*.

b. *Do Not Install All Below*

The *Install* indicators are deselected for the object in the selected hierarchy level and all objects in the lower levels of the hierarchy.

- ***Match (X) or Copy***

If the SAP delivery version and the active version can be matched, a checkbox is displayed in this column.

With the most important object types, the active version and the SAP delivery version can be matched.

Note that the InfoSource TRCS supports the match, but the 3.x InfoSource ISTD does not.

From a technical point of view, the SAP delivery version (D version) is matched against the M version. As in most cases the M version is identical to the active version (A version) in a customer system, this is referred to as a match between the D and A versions for reasons of simplification.

When a match is performed, particular properties of the object are compared in the A version and the D version. First it has to be decided whether these properties can be matched automatically or whether this has to be done manually. If you are sure that the object will be used in the same way after you install BI Content, you can perform an automatic match for those properties. When performing matches manually you have to decide whether the characteristics of a property from the active version are to be retained, or whether the characteristics are to be copied from the delivery version.

Example of an automatic match:

Additional customer-specific attributes have been added to an InfoObject in the A version. In the D version, two additional attributes have been delivered by SAP that do not contain the customer-specific attributes. To be able to use the additional attributes, the delivery version has to be reinstalled from BI Content. At the same time, the customer-specific attributes are to be retained. In this case, you have to set the indicator (X) in the checkbox. After installing the BI Content, the additional attributes are available and the customer-specific enhancements have been retained automatically. However, if you have not checked the match field, the customer-specific enhancements in the A version are lost.

Example of a manual match:

An InfoObject has a different text in the A version than in the D version. In this case the two versions have to be matched manually. When BI Content is installed, a details screen appears which asks you to specify whether the text should be transferred from the active version or from the D version.

The *Match* indicator is set by default to prevent the customer version being overwritten unintentionally. If the BI Content of the SAP delivery version is to be matched against the active version, you have to set the *Install* indicator **separately**.

The active version is overwritten with the delivery version if:

- The match indicator is not set, and
- The install indicator is set.

In other words, the delivery version is *copied* to the active version.

If the *Install* indicator is not set, the object is not copied or matched. In this case, the *Match* indicator has no effect.

In the context menu, two options are available:

a. *Merge All Below*

The object in the selected hierarchy level and all objects in the lower levels of the hierarchy are selected as to *Match*.

b. *Copy All Below*

The *Match* indicators are removed for the object in the selected hierarchy level and all objects in the lower levels of the hierarchy. If the *Install* indicator is also set, these objects are copied from the delivery version to the active version.

The most important properties that are taken into account when versions are matched are listed below.

When referring to InfoObjects it is important to differentiate between:

- Characteristics
- Time characteristics
- Key figures and
- Units

Valid for All InfoObjects Named Above:

Properties Matched Automatically:	**Properties Matched Manually:**
• Texts • Description If texts/descriptions in the delivery version do not yet exist in the active version, they are transferred.	• Attribute only (indicator) • Short and long text (description) • Time dependency of attributes • Transfer routine

Valid for Characteristics

Properties Matched Automatically:	**Properties Matched Manually:**

• Compounding New InfoObjects are included in compounding • Referenced characteristic Reference characteristics are transferred automatically. A warning is issued. • Attributes / navigation attributes New attributes are included. If an attribute is added by the customer, the settings for this attribute are transferred (for example, F4 sequence). • Application components If the entry for the active version is initial, the delivery version is used.	• BEx display • Characteristic constants • Default member • Authorizations (indicator and field) • Selection (CHASEL) • InfoObject is document attribute • Data type • Length • Output length • Conversion routine • Lower case allowed (indicator) • Contains master data (indicator) • Master data is authorization relevant (indicator) • Export DataSource (indicator) • F4 query design • F4 query runtime • Check DataStore object • Contains text tables (indicator) • Texts are language-dependent (indicator) • Short, medium and long texts exist (each with indicator) • Default currency • GIS attribute • Hierarchies exist (indicator)

Valid for Key Figures

Properties Matched Automatically:	**Properties Matched Manually:**
	• Type of key figure • Fixed unit • Fixed currency • Aggregation • Exception Aggregation

Valid for Units

Properties Matched Automatically:	**Properties Matched Manually:**

	• See the table above (*Valid for Characteristics*), if these properties are also relevant for the units.

When referring to InfoProviders it is important to differentiate between:

- MultiProviders
- InfoCubes
- DataStore objects and
- InfoObjects

InfoObjects are discussed above (see the first table *Valid for All Objects Named Above*).

Valid for All InfoProviders Named Above:

Properties Matched Automatically:	Properties Matched Manually:
• New InfoObjects If InfoObjects in the delivery version do not yet exist in the active version, they are transferred.	• Manual matching is not necessary.

Valid for MultiProviders

Properties Matched Automatically:	Properties Matched Manually:
• New InfoProviders If InfoProviders in the delivery version do not yet exist in the active version, they are transferred.	• Manual matching is not necessary.

Valid for InfoCubes

Properties Matched Automatically:	Properties Matched Manually:
• New dimensions If dimensions in the delivery version do not yet exist in the active version, they are transferred.	• Manual matching is not necessary.

With the InfoObject catalog object type, InfoObjects that are assigned to a catalog in the delivery version are transferred to the active version automatically. Manual matching is not necessary.

With the transfer rule object type, those objects that exist in the delivery version but are not yet contained in the active version (for example, additional attributes) are added automatically when the match is performed. Manual matching is not necessary.

With the process chain object type, those objects that exist in the delivery version but are not yet contained in the active version (for example, additional objects in the process chain) are added automatically when the match is performed. Manual matching is not necessary.

- ***Active Version Available***

  In this column, the following display options are available:

- : The object is available in an active version. You decide whether you want to retain this version or reinstall the latest version of the object.
- Date symbol: The object is already available in an active version. However, the active version belongs to an older BI Content release. We recommend that you install the latest version.
- Gray symbol : An activated version of the object is available in the system. However, the object status is inactive. If an object version is activated, this indicates that all metadata for the object can be created successfully. The inactive object status indicates that the processed object is inconsistent and cannot currently be used for processing data. Transfer the object again from the SAP delivery version (D version). If this is not enough to change the object status from “inactive” to “active”, the object has to be postprocessed. In the activation log, you can find notes on postprocessing.

  Postprocessing: An InfoObject may have status inactive if changes have been made to the structure at field level. Postprocessing the object involves converting the relevant tables in accordance with the activation log.
- No indicator: The object is **not** available in an active version.

7. Make Settings in the Selection List and *Install*.

Make the required settings in the *Install* selection list:

Installation Type

Installation Type	Information
Simulate installation	The system runs a test to see whether any errors are likely to occur during the installation. However, not all errors that may occur can be identified during simulation: Some error messages can only be generated when a real A version (not just a simulation) is available in the system. You should, therefore, only use the simulation function as a rough guide to help you identify and remove basic errors.
Install	The selected objects are installed immediately.
Install in Background	The selected objects are installed in the background.
Install and Transport	The selected objects are installed immediately and then written to a transport request.

We recommend that you use the *Install in Background* option because installing a large number of objects can take a long time. Modifying objects that are already active can take an especially long time. Make sure that when you install objects in the background, the versions are not matched. It is always the delivery version that is installed. They are not matched against any available customer versions.

8. Correct Errors

You can find out if errors occurred during the installation of BI Content by displaying a log. Choose the *Log* pushbutton to access the log. In the relevant maintenance transaction, repair any errors that occurred during the installation. Install the BI Content again.

See also:

Installing BI Content DataSources in Active Version [Page 524]

4.1.13 Customer and Partner Content

Purpose

Content that is delivered to business areas by an SAP NetWeaver Business Intelligence (BI) customer or consulting partner is referred to as customer or partner content. The functionality of customer or partner content complements and enhances the options for using the BI Content delivered by SAP.

The concept and technical implementation of customer and partner content largely correspond to one another. For more information about customer content, see Customer Content [Page 444]. For more information about partner content, see Partner Content [Page 503].

Integration

Understanding the documentation for the customer or partner content requires knowledge of the following topics due to the similarity between the two types of content:

- BI Content (Versions) [External]
- Transport System [External]

These topics describe standard functions without taking the features of customer or partner content into consideration.

4.1.13.1 Customer Content

Purpose

Customers of SAP NetWeaver Business Intelligence (BI) can develop their own content and deliver it to their business units. In the following text, we refer to these business units as BI *customers*.

BI customers can also make limited modifications and changes to BI Content. However, whenever new SAP BI Content is delivered, you have to compare the different versions manually.

Features

Customer content covers the following areas:

- Structure of customer-specific system landscape (see System Landscape [Page 445])
- Delivery of BI Content by SAP
- Development of customer-specific content by customers
- Changes in BI Content
- Delivery of customer content by customers to their own customers

Constraints

Namespace for Developing BI Objects

- You can use your own namespace (see Using Namespaces for Developing BI Objects [Page 450])

Note also Features when Transporting Roles [Page 474]. We recommend that you use the following schema when allocating the namespace: delivery namespace /ABC/ → customer namespace /ABC/C_. This corresponds to the SAP namespaces for roles (delivery namespace SAP_BW_ → customer namespace SAP_BWC_). However, this means that you cannot create roles that begin with the letter “C”.

Delivery of BI Objects

We recommend that you deliver customer content using the **Add-On Assembly Kits** instead of using transports, as described in the following documentation. Using this tool, you can combine the BI objects that you have developed into one add-on. Furthermore, you can specify installation requirements (such as particular SAP releases) in the systems of your customers. You can also simplify the maintenance of your delivered content by providing your own release and support package status for each delivery. This allows you to see which content version is installed in your customer systems.

The *Add-On Assembly Kit* is not part of SAP NetWeaver and must be purchased separately. No additional tools are required in your customers’ systems: The transactions for implementing your add-ons are available in all SAP systems.

- If you want SAP to certify your content, send an e-mail to the *Integration and Certification Center* (icc@sap.com). You can use this address to purchase the *Add-On Assembly Kit* together with an add-on certification contract.
- If you do **not** want your content to be certified, you can purchase the *Add-On Assembly Kit* from the *Add-On Assembly Kit Implementation Service* (AAKIS). For more information, see *SAP Service Marketplace* at `service.sap.com/slo` → *Repository Harmonization* → *Detailed Information* → *AAKIS*.

4.1.13.1.1 System Landscape

Purpose

You need at least one BI system for content development to deliver BI Content and develop your own content and deliver it as customer content. If you want to develop extractors and DataSources, you also need an OLTP system for content development. We recommend that you connect a content test system to each content development system. In the BI Content

test system, you check whether customer-specific content can be activated. You deliver a consolidated status from the test system to the OLTP and BI systems, for example, to your customer's central system, as the delivery transport.

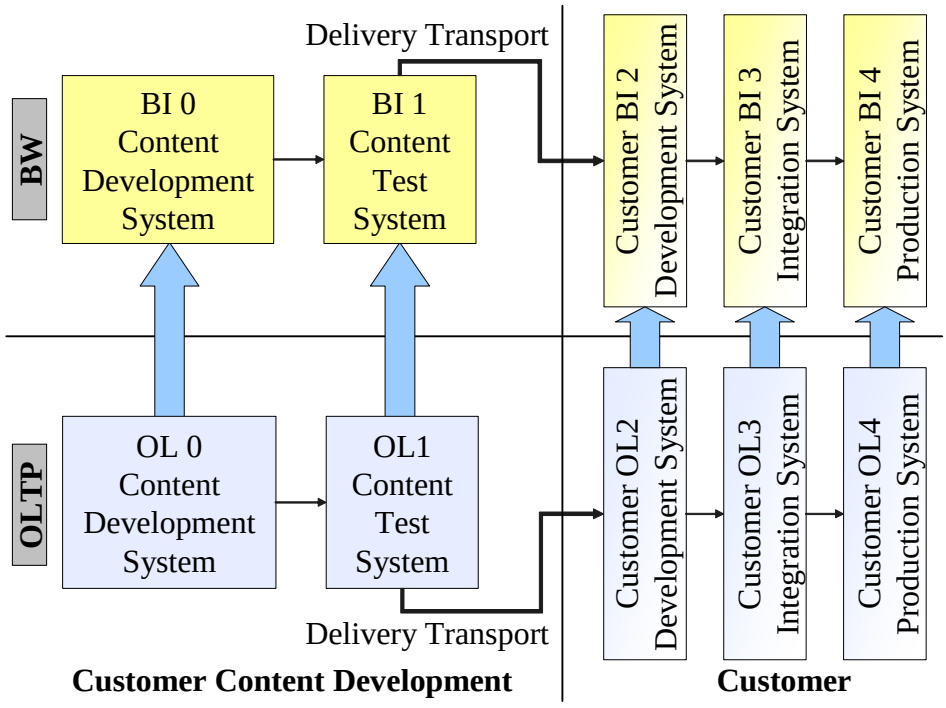
A content development system is an OLTP or BI system which is set as a content system in its system settings.

An OLTP content development system **must** have the lowest source system release of all the customer systems to which the customer wants to deliver content.

You can also set up a user for whom the system behaves as a content system by setting the appropriate parameters for this user.

For more information, see System Settings in OLTP Systems [Page 477] and System Settings in BI Systems [Page 447].

The following graphic shows the system landscape described:



Transport from the content test system and not from the content development system. Since only requests that have been released exist in this system, the objects are consistent, in contrast to objects in the content development system.

To deliver content, collect the content requests in the content test system and build one or more delivery transports from them. Merge the transport requests run in this system in the target request. The programs required to do this are available in the *Transport Organizer Tools* (SE03).

For more information, see Transport Organizer [External], in particular Transport Organizer (Extended View) [External].

The right side of the graphic normally shows a three-level customer system landscape to which BI customers deliver customer content. A system landscape of this type consists of one

OLTP and one BI system as the development, integration (or consolidation), and productive systems. You can further develop the landscape according to the needs of the customer, for example, by connecting a separate training system parallel to the production system.

4.1.13.1.2 System Settings in BI Systems

Purpose

Using special system settings, you can use a system as a content development system and switch user-specific system behavior on or off.

Process Flow

1. Activate all required BI Content objects with the BI-specific transport system. Active objects (A-TLOGO) are generally to be created in the $TMP package. For more information, see Installing BI Content [Page 435] as well as Standard and BI Transport Connection (A Comparison) [External].
2. Set the system setting for your BI system to content system.

   On the *Administrator Workbench: BI Content* screen, choose *Edit* → *Content System Administration*. The *Content System Administration* screen appears (transaction RSOCONTENT). Set the indicator for *System is a Content Development System*. Your BI system is now a content system.

   We recommend that you do **not** reset these system settings after having created objects.
3. We recommend that you activate the standard transport system so that new objects are automatically recorded. For more information, see Standard and BI Transport Connection (A Comparison) [External] as well as Switching the Standard Transport System On and Off [External].
4. If not all BI Content objects were activated or SAP has delivered new BI Content, you can reactivate the BI Content in your content system. Since the *BI Content* view is not active in a content system, you have to set up a specific user, for whom the system responds as if it were **not** a BI content development system. Furthermore, we recommend that you switch off the standard transport system for this user. In doing so, you ensure that objects are not inadvertently transported to the A version and delivered.
   a. Call user maintenance (transaction SU01). For more information, see Creating and Maintaining User Master Records [External].
   b. Assign the parameters RSOISCONTENTSYSTEM and RSOSTANDARDCTOACTIVE with a blank character as the parameter value to the user who is to activate the BI Content.

      For these parameters, the following is possible:

      - If the parameters are **not** entered, the system settings apply for the user.

- If the parameters are entered, the system settings do **not** apply for the user.

RSOISCONTENTSYSTEM: Depending on the value entered for this parameter, the system is either treated as a BI Content development system (parameter value: 'X') or is not treated as such (parameter value: blank character).

RSOSTANDARDCTOACTIVE: Depending on the value entered for this parameter, the standard transport system is either switched on (parameter value: 'X') or switched off (parameter value: blank character).

4.1.13.1.3 BI Content Delivery by SAP

Purpose

The standard procedure used in the delivery of BI Content by SAP is the model for the delivery of content by customers and partners. The basis of this process is BI object versioning:

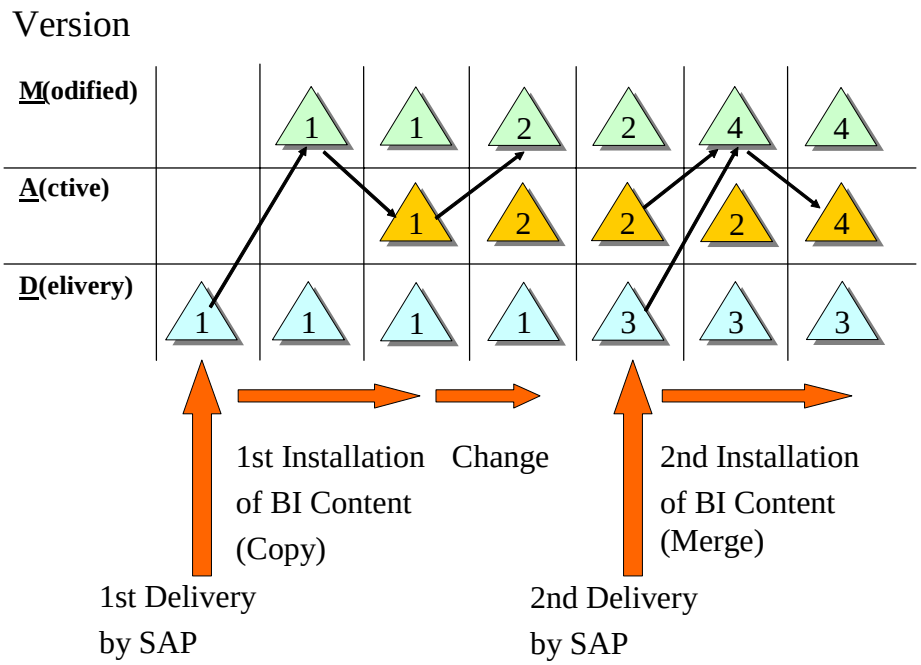
SAP delivers BI content objects in the SAP delivery version (D version). These objects are imported into the customers system in the D version.

If the customer changes BI Content objects, these are saved in the **M** version. You need to transfer these objects into the active version (**A** version) to be able to work with them.

For more information about object versions and delivery process of BI Content by SAP, see BI Content (Versions) [External].

Upgrading BI Content has **no** effect on BI objects that you already use productively, since only the D version is overwritten.

The following graphic shows how BI Content objects are used in a customer system with double delivery:



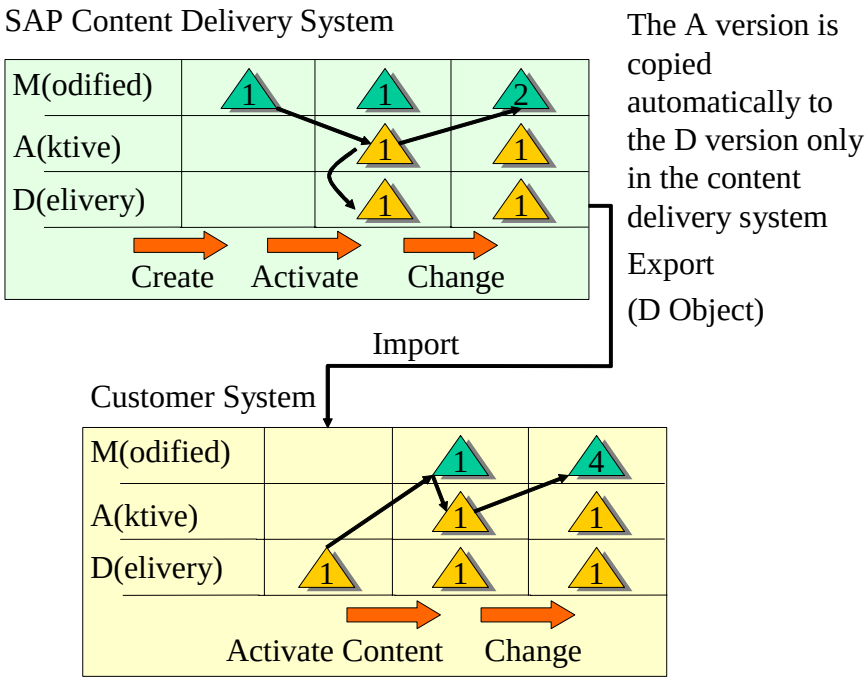
Process Flow

In the SAP content delivery system, BI objects are copied automatically to the D version when you activate them.

There are two logical transport objects for each BI object: an A and a D TLOGO object. In accordance with the system settings, the system writes the suitable TLOGO object to the transport request when transporting. When a content development system is involved, the system chooses the D TLOGO object.

For more information about transport-relevant metadata object types for BI, see Transport-Relevant Metadata Object Types [External].

The following graphic shows the delivery process for BI Content:



In the customer system, changes to BI objects are saved in the active version. However, they are **not** automatically copied to the D version when you activate them.

4.1.13.1.4 Using Namespaces for Developing BI Objects

Use

To avoid naming conflicts with a customer's local objects, customer content has to be developed in its own namespace. You can either use your own namespace or an agreed upon subarea of the standard customer namespace (A* to Z*):

1. Scenario: Your Own Namespace for Developing Customer Content

We recommend that you use your own namespace. Namespaces in the form /ABC are reserved for customer content development. To designate a BI object as having been developed in a namespace of this type, enter the technical name of the object together with the namespace.

For example, an InfoObject can have the technical name /ABC/MATERIAL.

For customers, using a special namespace for developing customer content is optional.

Partners **have** to develop their partner content in their own development namespace.

2. Scenario: Standard Namespace for Developing Customer Content

As a customer, you can also develop customer content in the standard customer namespace (A* to Z*). In this case, however, you have to make sure that the content objects and the recipient's personal objects are located in different subareas of your namespace.

You can split your standard customer namespace so that, for example, A* to D* is reserved for customer content development and E* to Z* is reserved for the development of personal objects.

In the source system note that for the application component hierarchy, the default customer namespace Y* and Z* is available.

The rest of this topic further explains how you can set up and use your own namespace for developing customer content, as detailed in the first (recommended) scenario.

Prerequisites

To apply for namespaces, you require an ABAP development license (ABAP/4 Development Workbench License).

For more information about applying for a namespace, see SAP Note 84282.

Procedure

1. Apply for a *development namespace* in the Service Marketplace at **`service.sap.com/namespaces`**.
2. Apply for a *generation namespace* as well as a development namespace. You apply for this in the same way as you apply for the development namespace.

   If you are using more than one development namespace, you require a generation namespace for each one.

   The name of the generation namespace must correspond to the following rules:

   - The name can have a maximum of seven characters, including a slash ("/").
   - The name has to start with the letter "B" and then consist only of numbers.

     /B2345/ or /B24/

   - The descriptive text starts with "BWGEN". Finally, the namespace to which it is assigned is specified.

     "BWGEN generation namespace for /ABC/ company ABC ...".

   The shorter the name of the generation namespace, the more significant characters you have available for the technical name of the BI object.
3. Release the development namespace and the generation namespace in BI and the OLTP system (Use the same namespaces in both systems).

   To do this, enter the namespace and the repair license in the namespace table in your development system. From the *Transport Organizer Tools* (transaction SE03), choose *Administration* → *Display/Change Namespaces*. The *Display View "Repository Namespaces ": Overview* screen appears. (In the OLTP system, path details may differ from release to release.)

Unlike the development namespace, you only enter the generation namespace with role **C** (customer). Therefore, you only need a repair license.

You can **only** create generated objects in the generation namespace. If you create your own, nongenerated objects in the generation namespace, there is a danger that these may be overwritten by objects generated in the BI system. For this reason, always use the development namespace for nongenerated objects.

4. Enter the namespace in your BI system.

   You can edit BI namespaces in transaction *Maintain BI Namespaces* (transaction RSNSPACE). Since this transaction uses a transport connection, your entries are written to a request and can therefore be transported to a target system. The system writes partner and customer namespaces such as /ABC/ to the RSPSPACE table. The RSNSPACE table contains the BI standard namespaces such as /BI0/ and /BIC/.

   In the OLTP system, set the namespaces to *changeable*. You make this setting in *Transport Organizer Tools* (transaction SE03) *Administration* → *Set System Change Option*. The *System Change Option* screen appears. Here you can set the namespaces to *modifiable.* (In the OLTP system, path details may differ from release to release.)

5. When you use a namespace, you should create a package with the same namespace prefix (such as /ABC/PACK).

Result

You have created a development namespace and a generation namespace.

Note that the length of the names of BI objects that are located in these namespaces is restricted. The following table contains some examples. Prerequisite: These examples are based on the assumption that your development namespace has *N* characters and your generation namespace has *G* characters. (Note that the slashes count as well, for example namespace /ABC/ has five characters).

Maximum Length of Names of BI Objects

BI Object Type	Maximum Length According to Namespace
InfoObject	The maximum length corresponds to the smaller of either: 14 characters - *G* or 14 characters - *N*. If you are not using analysis authorizations (see Authorizations [External]) but are using the reporting authorizations of the previous authorization concept, the following length restrictions also apply for authorization-relevant characteristics: With authorization-relevant characteristics /ABC/NAME, the field name for the authorization fields must **not** have more than 10 characters. If the field name of characteristic /B24/<NAME> is longer, you have to define a shorter authorization field name explicitly in InfoObject maintenance. For more information about the previous concept of reporting authorizations, see Previous Concept for Reporting Authorizations [External].
InfoCube	14 characters – *G*
DataStore object	13 characters – *G*

InfoSource	25 characters – *G*
InfoSet	14 characters – *G*
DataSource	30 characters – *G*

Note that existing BI objects **cannot** automatically be copied into the namespace. To put an existing object in the namespace, you have to create a new object in the namespace as a copy of the required object.

See also:

Features when Transporting Roles [Page 474]

4.1.13.1.5 Object Changeability in Customer Content Systems

Use

Since changes to objects from SAP BI Content require matching and thus require manual efforts, we recommend that you do not change objects if at all possible. However, if you want to change SAP objects, we recommend that you limit this to a few important objects (see Content Delivery by Customers [Page 454] and InfoAreas, Application Components, and InfoPackage Groups [Page 475]).

To prevent customer changes from getting lost with a new delivery of BI Content, the BI Content system does not allow objects delivered by SAP to be changed as a default. The following exceptions to this rule exist in a customer content system.

Object Changeability in Customer Content Systems

The following objects can be changed in a customer content system:

- Objects in the namespace of the role **P** (*Producer*)
- Objects in standard customer namespace (A* to Z*)

Changeability for Other Objects

You can set other objects imported into your system to changeable by making entries in *Content System Administration* (see *Activities*).

Prerequisites

You have set the customer development system as a content development system. For more information, see System Settings in BI Systems [Page 447].

Activities

1. On the *Administrator Workbench: BI Content* screen, choose *Edit* → *Content System Administration*. The *Content System Administration* screen appears (transaction RSOCONTENT).
2. Choose *Switch External Content Objects to Changeable*. The *Change View "Changeable Objects in Customer/ Partner Content System"* screen appears.

3. Enter the objects to be changed with their *Object Type* and *Object Name* in the table. Generic entries (with */?) are also possible.

   For example, if you want to change all objects that begin with the character string 0AB to changeable, enter 0AB* into the *Object Name* field.

4. Choose the mode of changeability. The following mode are available:
   - *Object Cannot Be Changed*. This option is the recommended default setting.
   - *Object Can Be Changed* (*But Not in D Version*) *No CTO Connection*. Only use this option for testing purposes, for example, if you want to set an object to changeable, without transporting it immediately with the Change and Transport Organizer (CTO).
   - *Object Can Be Changed* (*In D Version Too*) *Has CTO Connection*. To be able to set objects to changeable and to transport them in exceptional cases, we recommend this option.

4.1.13.1.6 Content Delivery by Customers

Purpose

Scenario 1

The simplest scenario consists of the following processes:

- You develop your own content.
- You do not use BI Content delivered by SAP or you use it without changing it.
- You deliver only your own content to your customers.

Scenario 2

You also change the BI Content delivered by SAP and deliver the modified BI Content to your customers.

We recommend that you do not change any, or you change as few objects as possible (see Object Changeability in Customer Content Systems [Page 453]).

Prerequisites

You have set the customer system as the content development system (see System Settings in BI Systems [Page 447]).

You have installed all required SAP BI Content objects from the content using the special user (see System Settings in BI Systems [Page 447]).

Process Flow

Scenario 1: Developing Your Own Customer Content

To deliver customer content you have developed, you proceed as follows:

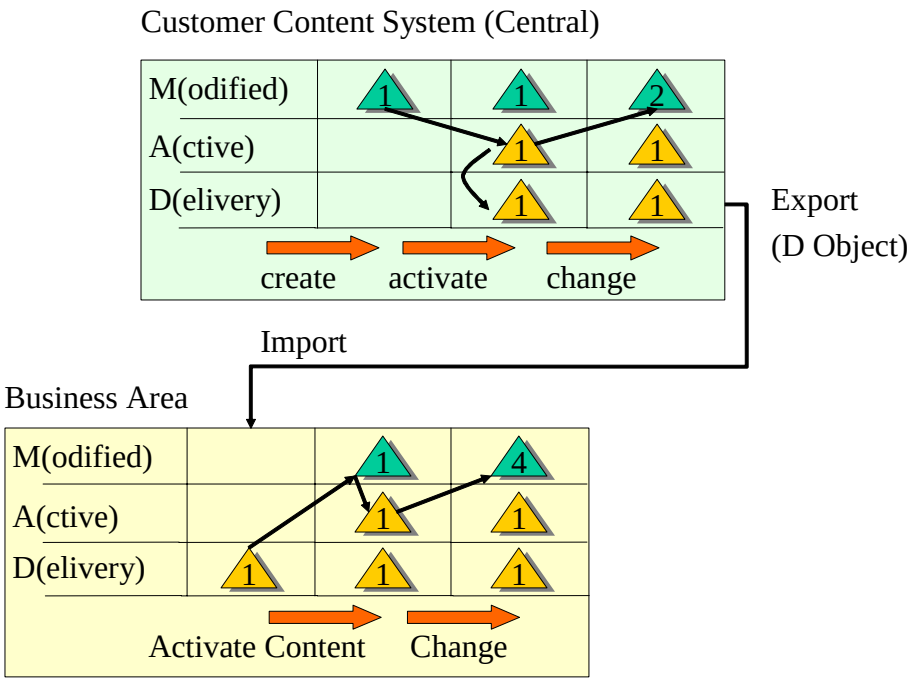
1. Create the required objects in your content development system and activate the objects that are first created in the M version (first M version → first A version). The system copies the activated objects automatically to the D version (first D version).

   If you want to transport the objects that you created before you set the system as the content system in your BI system settings, you simply activate them by copying them to the D version.

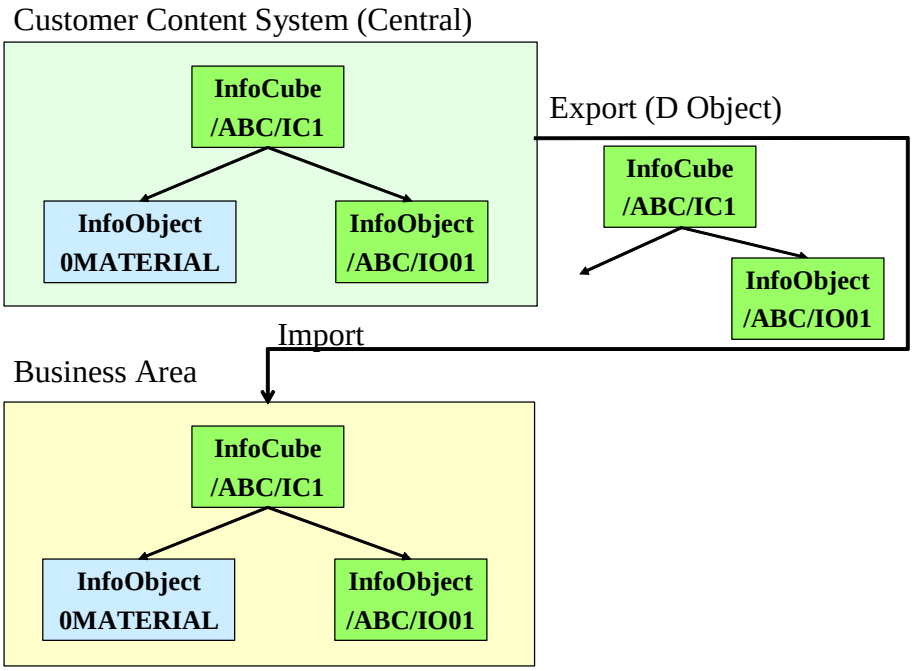
2. Since the standard transport system is switched on, the system wrote your content objects to transport requests. Release these requests. Your content objects are transported from the content development system to the content test system.

3. Deliver the objects in the D version from the content test system using a delivery transport (see System Landscape [Page 445]). Make sure that no A TLOGO objects are transported. Further changes (second M version) do **not** affect delivery as long as you do not activate these objects.

The following figure illustrates this procedure:



The following figure illustrates the delivery of customer content that was developed in a separate namespace /ABC/ (see Using Namespaces for Developing BI Objects [Page 450]). The objects are exported in the D version. The 0MATERIAL InfoObject is used but not delivered.



Scenario 2: Adjusting BI Content

It is often more effective to use or adjust BI Content delivered by SAP instead of fully creating your own content. You can deliver adjusted BI Content to your customers as part of customer content.

The delivery process flow corresponds to the delivery process flow described in scenario 1. However, you also need to set any SAP objects that you want to adjust to changeable in table RSOCONTCHNGABLE (see Object Changeability in Customer Content Systems [Page 453]).

Delivering Modified BI Content After Installing an SAP Support Package or After an Upgrade

Note that after upgrading or installing an SAP support package, changes that you made as a customer to any BI objects delivered by SAP are lost in the D version if SAP delivers this object again.

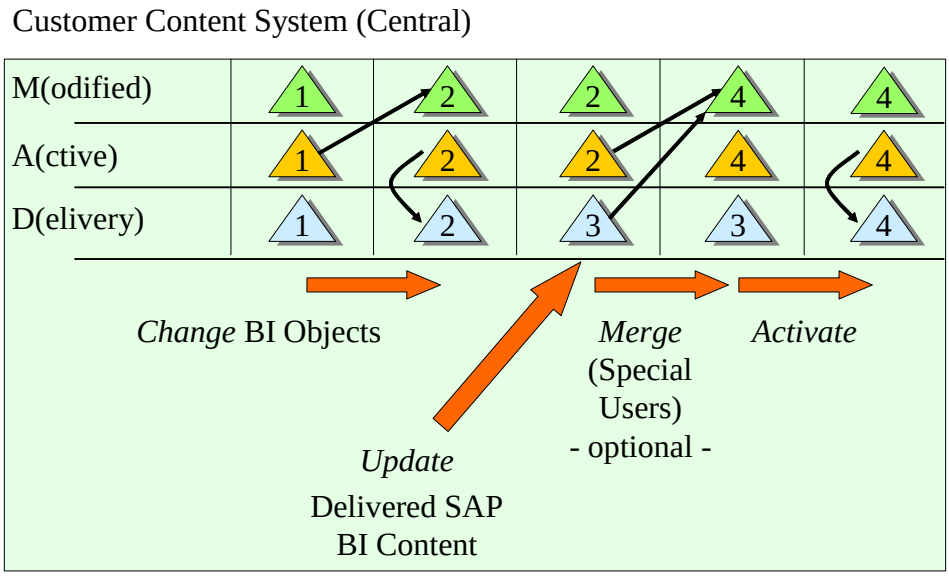
We recommend the following procedure:

a. Install the SAP support package in your content development system.

b. Create a correction delivery transport.

c. When upgrading other systems, include the correction delivery transport in the upgrade.

The following figure illustrates each of the steps in the recommended process flow:



The process flow in the figure outlines the following steps:

1. Activate all BI Content objects delivered by SAP (first A version).
2. Make the required changes (second M version).
3. Activate the changed objects (second A version). The system copies the activated objects automatically to the D version (second D version).
4. You can deliver your customer content in the D version (second D version).
5. If SAP redelivers BI Content in a support package or in an upgrade (third D version), you have to restore your objects in the D version.

   The system indicates that certain BI Content objects were modified and need to be merged. The modified BI Content objects are listed hierarchically in the overview of objects for the modification adjustment (transaction SPAU). (Within the system, table SMODILOG is used to do this; this table is also transported to the subsequent systems.)

   Alternatively, in the *BI Content* functional area of the Data Warehousing Workbench, you can select the *Objects in BI Patch* view and check which objects SAP is delivering in the current support package. For upgrades, it is usual for most objects to be redelivered.

   To restore your objects in the D version, you can proceed in various ways:

   - **Without Merging**

     Customers generally only need their own version (second version). All that remains to be done is to reactivate the objects (using a normal user). This automatically overwrites the D version delivered by SAP (the third D version). However, with this procedure, all enhancements delivered by SAP are lost.

   - **With Merging**

If you want to keep enhancements delivered by SAP, you can merge the new content version (third D version) with the active version (second A version).

Merge the versions as required and add the objects to a transport request. If you use the Add-On Assembly Kit (AAK), you can create a *Conflict Resolution Transport* in your consolidation system with minimal effort (see Customer Content [Page 444]). Otherwise you can put the request together manually in the content development system.

The following table provides an overview of the object types for which merging is automatically available. If automatic merging is **not** supported, you have to manually compare the two versions (second A version and third D version) and adjust them if necessary.

Availability of Automatic Merging for Object Types

With Automatic Merging	Without Automatic Merging
Report-report interface, RRI recipient of the InfoCube, RRI recipient of query	Aggregate
DataSource/FileSource	Application component
InfoCube	Crystal Report
InfoObject	Query element: query, structure, restricted and calculated key figures, variable
InfoObject catalog	Update rules
Data Mining: mining model, model source	InfoArea
MultiProvider	InfoPackage
DataStore object	InfoPackage group
Process chain	InfoSet, classic InfoSet
Transfer rules	InfoSet query
InfoSource TRCS	3.x InfoSource ISTD
	Source system
	Query view
	Reporting Agent: Reporting Agent scheduling package, Reporting Agent setting
	Role
	Currency translation type
	Web item
	Web template
	Workbook

Note that you need to merge the versions using the preferred user for which the system is **not** a content system and the standard transport system is switched off (see System Landscape [Page 445]). If objects are not to be written to the

$TMP package but to another transportable package, the requests for this user are **not** to be released and transported.

Finally, you have to reactivate the content using a normal user, so that the objects are copied to the D version. If no changes were made and there is nothing to be activated, you may need to make a pseudo change to the objects.

6. When activated, the objects are written to one or more transport requests. Transport the transport request(s) to the test system, and use it to create a correction delivery transport. If you use the Add-On Assembly Kit (AAK), you can create a *Conflict Resolution Transport* in your consolidation system with minimal effort (see Customer Content [Page 444]).
7. Deliver your correction delivery transport or the *Conflict Resolution Transport* directly after installing the SAP support package or after an upgrade on all subsequent systems.

4.1.13.1.7 Transport and Delivery of Source-System Dependent Objects

Use

To facilitate understanding of this topic, this section contains information on transports in general.

In BI systems, objects are not transported, but rather the metadata that is required to generate an object in the target system. Source-system dependent objects are notably dependent on circumstances in the target system. They can be distributed, even physically, to a BI system and to a source system. As a result, particular requirements often exist in relation to source-system dependent objects, such as the following:

- An object that was created in the original system for a source system A, is to be created in the target system for another source system B.
- Since the definition of a connection to a source system always depends on both sides, the source system and the BI system, an object that was created in both the source and target systems for one and the same source system physically has two different connections to this system.

Therefore, with source-system dependent objects, a logic runs that defines the name of and connection to the source system for which the import and activation of the content is to be executed.

Delivery (Content-Specific)

As with the delivery of SAP BI Content, the name of the source system is normally of no relevance with the delivery of customer content. This is because the customer has other source systems connected than the content development system. Therefore, source systems are made anonymous before the delivery of source-system dependent objects and replaced by their release status. New objects for SAP NetWeaver 2004s, DataSources, transformations, and DTP, are replaced by the DataSource version. This means source-system dependent objects need to be delivered as individual object types, such as *shadow objects.* These carry the release or version instead of the source system in their key.

Transport (General)

Since the key (name) of source-system dependent objects in the target system differs from the key for the object in the original system, metadata cannot be imported in the same version as it exists in the original system (A or M version). There could be other objects in the target system that have the same key as the object in the original system. Therefore, source-system dependent objects are transported in their own version, the T (transport) version.

After-import methods operate on the T version and generate the modified and active version from this.

For more information, see Transport Systems [External].

Features

The following table provides an overview of the different types of source-system dependent objects:

Source-System Dependent Objects

Object Type	Example
Source-system dependent object that carries the source system directly in its key.	DataSource [Page 460] Transfer Rule (SAP BW 3.x) [Page 466] RRI recipient for an InfoCube RRI recipient for a query
Source-system dependent object that carries the source system in its shortened key according to a hash algorithm.	Transformation [Page 463]
Source-system dependent object that carries the source system in its key in coded form.	Transfer Structure (SAP BW 3.x) [Page 468]
Source-system dependent object that carries a GUID [External] (*Globally Unique Identifier*) in its key.	Data Transfer Process [Page 463] InfoPackage [Page 464]
Source-system independent object that refers to source-system dependent objects.	Process chain Process chain variant (PSA processes) See: Process Chains and Process Variants [Page 470]

In the first four cases, the keys change. In this way, a distinction can be made between their D and A tables.

4.1.13.1.7.1 DataSource

Definition

A **DataSource** is a source-system dependent object that carries the source system directly in its key.

Use

The following types of DataSouce objects exist:

- Active object: **RSDS** / delivery object: **SHDS**: This DataSource is available as of SAP NetWeaver 2004s. If you work with object concepts changed in SAP NetWeaver 2004s in data flow and process design (transformation, InfoPackage for loading into PSA, data transfer process for data distribution within BI), use this DataSource to transfer data from a source system to BI.
- Active object: **ISFS** / delivery object: **SHFS**: This DataSource is available as of SAP BW 3.x and is called 3.x DataSource. If your dataflow is modeled using objects that are based on the previous concept (3.x InfoSource, 3.x transfer rules, 3.x update rules) and the process design is built on these objects, you can continue to work with 3.x DataSources.

For more information about these BI objects and their limitations, see DataSource [Page 20].

Structure

Key Fields of DataSource (RSDS / SHDS)

Version	Key Fields
A (active) / M (modified) / T (transport) – RSDS	DataSource, source system, object version
D (delivery or content) – SHDS	DataSource, version type, version

In the D version, *version type* denotes the content release type, and *version* denotes the content version. Using the content properties, content DataSources and the appropriate content transformations are assigned to each other and to the source system.

For more information, see Content Properties of a DataSource [Page 462].

Since the version is compounded to the DataSource, there should always be a suitable target source system for content delivery. The D version can therefore be created 1:1 in the content system.

Key Fields of 3.x DataSource (ISFS / SHFS)

Version	Key Fields
A (active) / M (modified) / T (transport) – ISFS	DataSource, source system, object version
D (delivery or content) – SHFS	DataSource

Since a 3.x DataSource (SHFS) is delivered in the BI system only for file or BAPI systems, the release (in contrast to a transfer rule (SAP BW 3.x) [Page 466]) for this object is always initial. Therefore, the release of the source system as individual field is missing in the key.

For more information about the delivery of DataSources in OLTP systems, see Delivery and Use of Customer Content in OLTP Systems [Page 482].

Integration

Note that the system always uses previous logic for a DataSource of type RSDS / SHDS whenever content development does **not** assign a content version. Therefore, the system evaluates the source system release in the D version of the DataSouce as a key field.

4.1.13.1.7.1.1 Content Properties of a DataSource

Use

On the DataSource Maintenance screen, on the *General Info.* tab page, you can specify the *Content Release Type* and the *Content Version* as content properties of the DataSource.

The content release type describes the type of content version. Combined with the content release type, the content version defines the relevancy/usability of a content object for a specific source system or the assignment of the object for a specific release-dependent content scenario.

Using the content properties, content DataSources and the suitable content transformations are assigned to each other and to the source system.

Features

The following content release types are available:

R/3

R/3 consists of DataSources for which the content version corresponds to the Basis or the SAP NetWeaver release of the SAP source system, in which the DataSource exists. DataSources with this type were developed before PI Basis 2005.1 and therefore have no explicit version. Suitable transformations also have R/3 as a content release type and their content version (the Basis or SAP NetWeaver release) is smaller or the same size as the Basis or SAP NetWeaver release of the source system. If there are multiple objects with various versions, the object with the largest version is used.

SAP

These are SAPI DataSources which are given an explicit version in BI Content development.

The version is independent of the SAP system release in which the DataSource was developed. The version mirrors the development status of the special DataSource. Suitable transformations are those which match the content release type and content version with the type and version of the DataSource.

Other Content Release Types

A type and release are to be assigned to non-SAP source systems. When content is created, DataSources and transformations inherit this type and release as content release type and content version. Suitable objects for a source system have the same content release type as the source system; the content version is the same size or smaller than the release of the source system. If there are multiple objects with various versions, the object with the largest version is used.

4.1.13.1.7.2 Transformation

Definition

If a **transformation** (active object: TRFN / delivery object: DTRF) uses a DataSource (RSDS / SHDS) as a source, it is a source-system dependent object that carries the source system in its shortened key according to a hash algorithm.

In contrast to the coding of the source system for a transfer structure (SAP BW 3.x) [Page 468], for which the encryption information is in a separate cross-classified table, the hash algorithm only requires the name of the source system for generating the hash key.

Structure

Key Fields of a Transformation (TRFN / DTRF)

Version	Key Fields
A (active) / M (modified) / T (transport) – TRFN	Tran ID
D (delivery or content) – DTRF	Tran ID

The Tran ID is a 40 byte hash key for combining source and target. Tran IDs are generated for different versions in the same manner. They differ in that the RSDS name of the DataSource is used in the first case, and the SHDS name of the DataSource is used in the D version.

The transformation is compounded to the DataSource using the content version. The transformation implicitly has a content version. In the after-import processing, a delivered transformation in a specific content version generally becomes several source-system dependent D versions. The source systems of these D versions are determined from existing source-system dependent DataSources that match the content version of the transformation.

See also:

DataSource [Page 460]

Data Transfer Process [Page 463]

4.1.13.1.7.3 Data Transfer Process

Definition

If a data transfer process (active object: DTPA / delivery object: DTPD) uses a DataSource (RSDS / SHDS) as a source, it is a source-system dependent object that carries the GUID in its key.

For more information about this BI object, see Data Transfer Process [Page 320].

Structure

Key Fields of a Data Transfer Process (DTPA / DTPD)

Version	Key Fields	Data Fields	Original Reference
A (active) / M (modified) / T (transport) – DTPA	GUID2	Source system	GUID1
D (delivery or content) – DTPD	GUID1	Version type, version	<blank>

The data transfer process is compounded to the DataSource using the content version. The data transfer process implicitly has a content version. In the after-import processing, a delivered DTP in a specific content version generally becomes several source-system dependent D versions. The source systems of these D versions are determined from existing source-system dependent DataSources that match the content version of the DTP.

See also:

DataSource [Page 460]

Transformation [Page 463]

4.1.13.1.7.4 InfoPackage

Definition

An **InfoPackage** (active object: ISIP / delivery object: SHIP) is a source-system dependent object that has the GUID in its key. There is one object for each source system.

For more information about this BI object, see Maintaining InfoPackages [Page 343].

Structure

Key Fields of an InfoPackage (ISIP / SHIP)

Version	Key Fields	Data Fields
A (active) / M (modified) / T (transport) – ISIP	GUID1	Source system
D (delivery or content) – SHIP	GUID2	Source system release

Since the source system does not exist in the key, but is contained within the data part, the InfoPackage object is also delivered as a shadow table.

Example

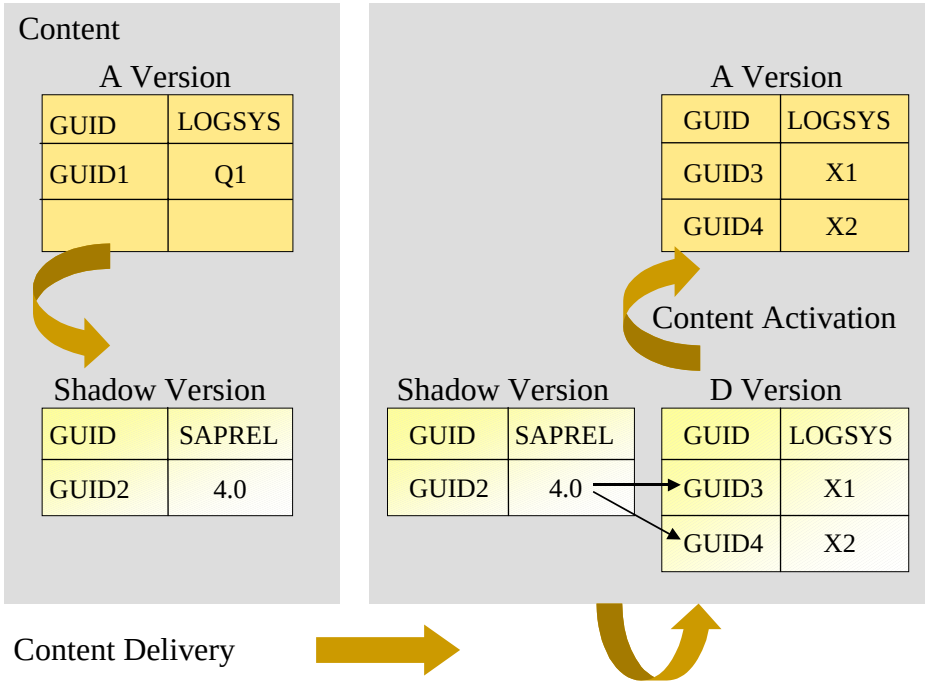
The following example shows what you need to bear in mind when delivering InfoPackages using customer content.

The content development system is connected to the Q1 source system. There is exactly one InfoPackage with GUID1 for this source system.

In the customer system, this InfoPackage is to be activated for several source systems with the same release status:

- Source system X1, release 4.0
- Source system X2, release 4.0

Accordingly, the InfoPackage is copied twice to the A version. The logic corresponds to the logic for transfer rules (see Transfer Rule (SAP BW 3.x) [Page 466]). Conversion is carried out by the **RSSM_SHIP_PSEUDO_DVERSION_WIR** function module. The copies get GUID3 and GUID4.



When SAP delivers again, you have to know from which InfoPackage GUID3 and GUID4 are derived. This information is stored in the RSSHIPDVERS table:

RSSHIPDVERS Table

LOGDPID_SH	LOGSYS	LOGDPID
GUID2	X1	GUID3
GUID2	X2	GUID4

Using the RSSM_SHIPDVERS_CLEANUP report, you can display and repair inconsistent InfoPackages if necessary. For more information, see SAP Note 844739.

See also:

Transport and Delivery of Source-System Dependent Objects [Page 459]

4.1.13.1.7.5 Transfer Rule (SAP BW 3.x)

Definition

Transfer rules (active object: ISMP / delivery object: SHMP) are source system dependent objects that carry the source system directly in their keys.

For more information about these BI objects, see Processing Transfer Rules [Page 281].

Structure

Key Fields of a Transfer Rule (ISMP / SHMP)

Version	Key Fields
A (active) / M (modified) / T (transport) – ISMP	DataSource, source system, object version
D (delivery or content) – SHMP	DataSource, source system release

The T version is used for a normal transport. The source systems are converted by the RSLOGSYSMAP table.

When transporting content objects, note that transfer rules (SHMP) are the only content objects in the InfoSource area that have an after-import method. This after-import method 'multiplies' the release-dependent shadow table (**not** the source-system dependent shadow table) and finds the best content release for each source system. Content is generated for each source system in this manner. The same logic runs when a new source system is connected. Content appropriate for this system is likewise generated from the shadow table.

This multiplication is carried out by the **RSAOS_PSEUDO_DVERSION_UPD** function module

The content version of an object subsequently caries the source system in its key. When the content is activated, the objects for this source system can be activated.

Example

The following example aims to show what you need to pay attention to, as far as transfer rules are concerned, regarding customer content and the delivery of source-system dependent objects having the source system in their keys.

The content development system is connected to the following source systems:

- Source system Q1, release 3.1
- Source system Q2, release 4.5
- Source system Q3, release 4.5

The customer target system is connected to the following source systems:

- Source system X1, release 3.1
- Source system X2, release 4.0

The left side of the following graphic shows the conversion from the A version to the shadow version in the content development system. The result of this conversion is that the source system is replaced by its release. Since Q2 and Q3 have the same release, the shadow version only contains two data records.

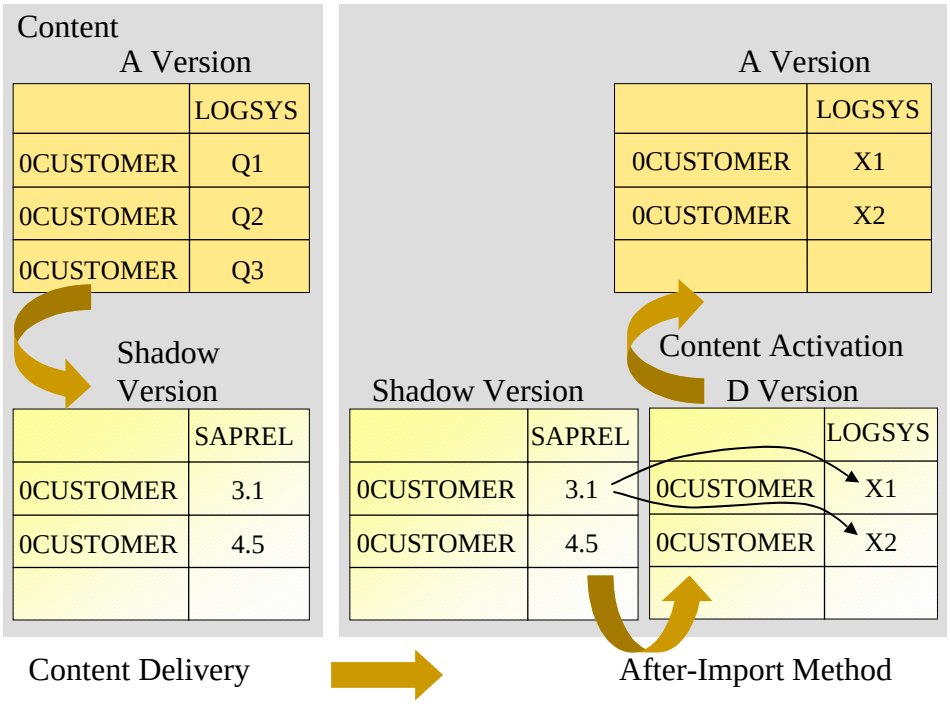
The shadow table is automatically generated when this is activated.

Avoid using the same DataSource with the same names, in different source systems that have the same release. In this case, only **one** delivery object is generated in the shadow tables that is combined from the information in different source systems. If this leads to a conflict, the most recently activated object has priority. This is also true for the Transfer Structure (SAP BW 3.x) [Page 468].

We recommend that you set up a customer content development system for each customer release.

The shadow version is delivered 1:1. The right side of the graphic shows the process flows in the customer system. The D version is generated from the imported shadow version in the first after-import step. The system searches for the appropriate release in the content for source system X1 and finds 3.1. Since there is no release 4.0 in the content, the system takes the next lowest release for source system X2, which is also 3.1. When activating the content, the D version is copied to the A version.

As a customer, code upwards compatible (additive).



See also:

Transport and Delivery of Source-System Dependent Objects [Page 459]

4.1.13.1.7.6 Transfer Structure (SAP BW 3.x)

Definition

The **transfer structure** (active object: ISTS / delivery object: SHTR) is a source-system dependent object. It does **not** carry the source system directly, but uses a transfer structure prefix that is coded in its key.

For more information about this BI object, see Transfer Structure in Data Flow 3.x [Page 555].

Structure

Key Fields of a Transfer Structure (ISTS / SHTR)

Version	Key Fields
A (active) / M (modified) / T (transport) – ISTS	DataSource with coded connection, object version
D (delivery or content) – SHTR	DataSource, source system release

Since a transfer structure (A / M version) carries a transfer structure prefix in its key, its name is replaced with the name of the transfer rule for which it is valid before delivery (virtual key). As a result, a transfer structure in the D version also has the release in its key.

More detailed information about this conversion is given in the example below.

Since the target tables in the customer system do **not** carry the source system in a key, the shadow tables of the transfer structure are **not** physically multiplied, as those in the transfer rule are.

With a transfer structure, this ‘multiplication’, (the replacement of the release by the respective source system), takes place while the objects are collected for content activation. The object collector then operates with virtual transfer structure names that have no physical correspondent.

The system finds the transfer structure prefix, and with it the new name for the transfer structure, by identifying the source system. When the system reads the transfer structure appropriate for this source system from the content, it takes the transfer structure with the next lowest release to the release for the source system.

Example

The following example aims to show what you need to paid attention to when delivering transfer structures within customer content.

The content development system is connected to the following source systems:

- Source system Q1, release 3.1
- Source system Q2, release 4.5
- Source system Q3, release 4.5

The transfer structure **does not** have the source system (field LOGSYS) in its key, but the name of the DataSource to which transfer structure has been added. The RSBASIDOC table shows to which source system, in which release, a transfer structure prefix is assigned.

RSBASIDOC Table

SLOGSYS	SAPREL	TSPREFIX
Q1	3.1	QA
Q2	4.5	QB
Q3	4.5	QC

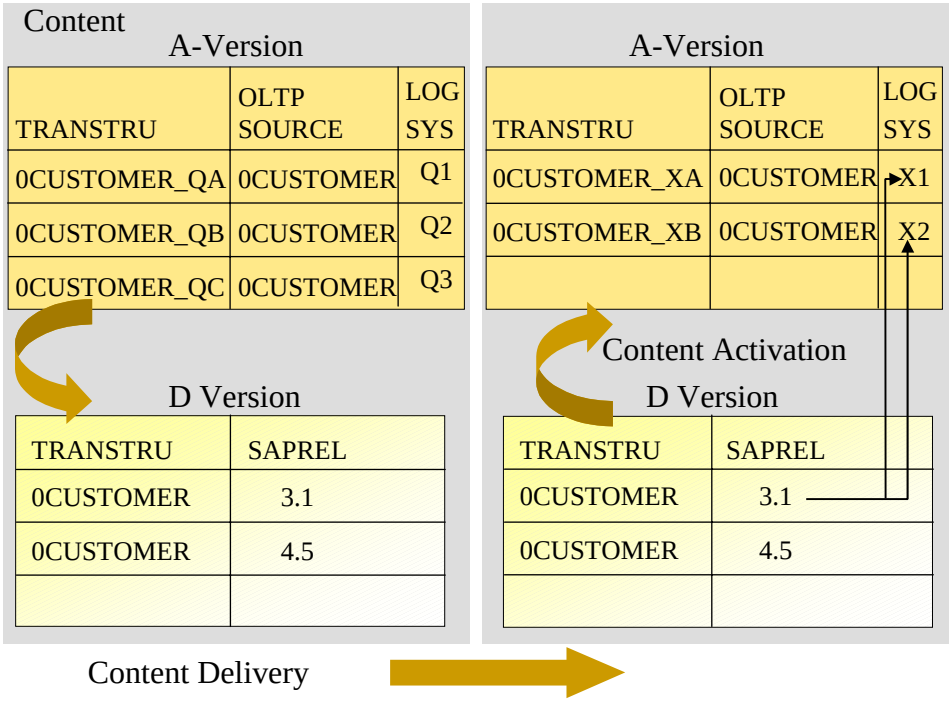
The customer target system is connected to the following source systems:

- Source system X1, release 3.1, transfer structure prefix XA
- Source system X2, release 4.0, transfer structure prefix XB

The left side of the following graphic shows the conversion from the A version to the D version in the content development system. Note that only the first column (TRANSTRU) is a key field. The system writes the name of the DataSource (OLTPSOURCE) to the TRANSTRU field in the D version and determines the release of the source system (SAPREL) from the transfer structure prefix. As a result, the name of the transfer structure is replaced in the D version with the name of the transfer rule for which the transfer structure is valid. Since Q2 and Q3 have the same release, the D version only contains two data records.

Avoid using the same DataSource with the same names, in different source systems that have the same release. For more information, see the notes in Transfer Rule (SAP BW 3.x) [Page 466].

The D version is delivered 1:1. The conversion on the right side corresponds, in principle, to that of the transfer rule. However, it actually shows the features described above (virtual transfer structure names).



See also:

Transfer Rule (SAP BW 3.x) [Page 466]

Transport and Delivery of Source-System Dependent Objects [Page 459]

4.1.13.1.7.7 Process Chains and Process Variants

Definition

Process chains (active object: RSPC / delivery object: DSPC) and **process variants** (active object: RSPV / delivery object: DSPV) are **not** source-system dependent objects. However, they can reference source-system dependent objects.

For more information about these BI objects, see Process Chain [Page 375] and Process [Page 377].

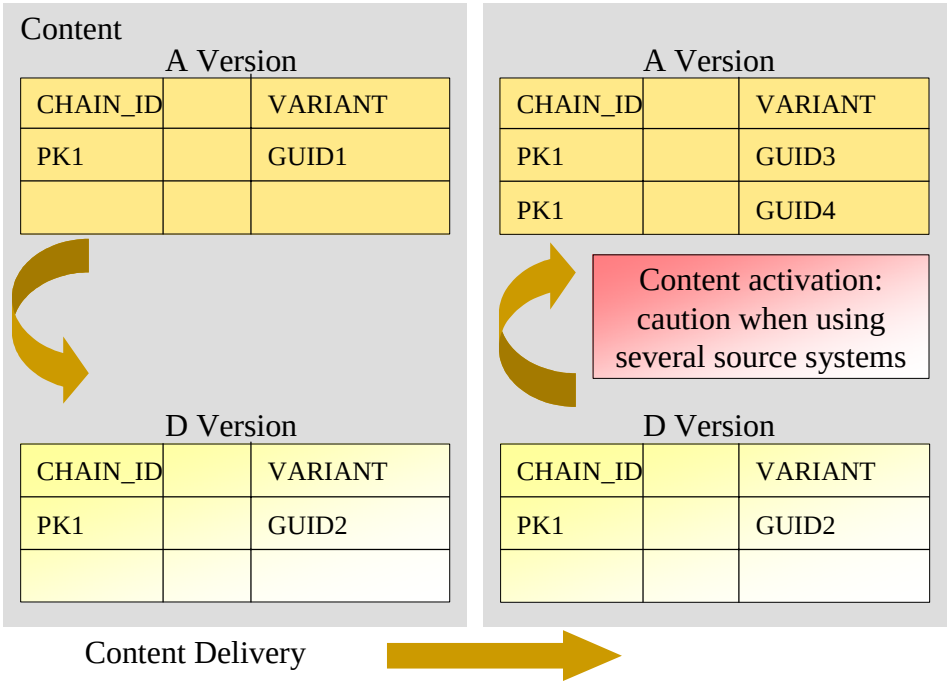
Structure

Process chains and process variants are normal transport objects. The key of these objects is the same key in the target and original system. However, process chains and process variants can reference source-system dependent objects.

A process chain that references source-system dependent objects (which have a GUID in the key) has this GUID in the VARIANT data field (see graphic).

When content is activated, the keys of these fields are converted in the same way as the corresponding source-system dependent object. The same differences exist between the A and D versions of the process chains and process variants.

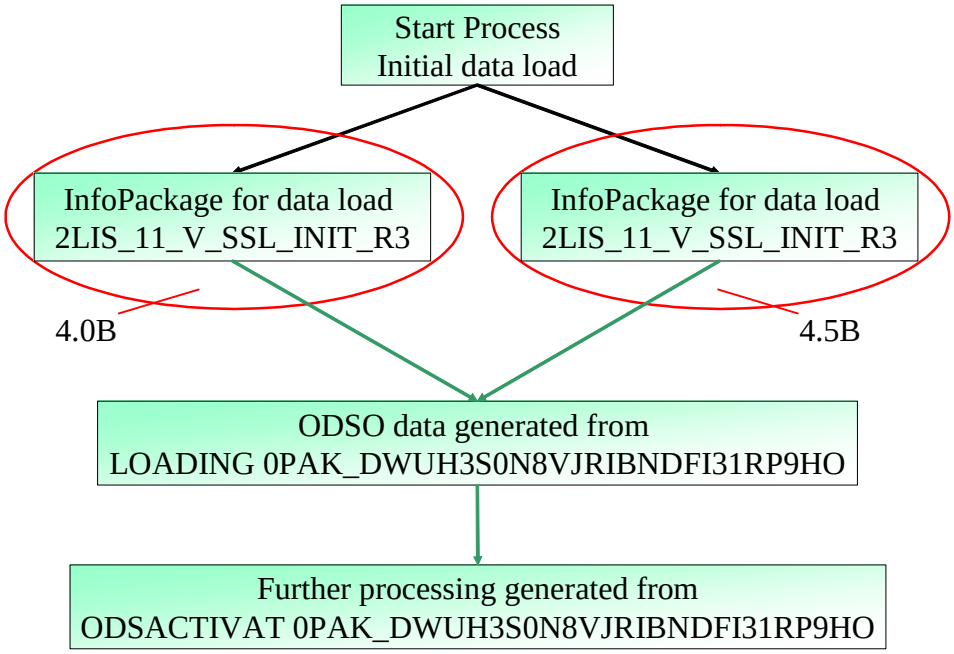
The following graphic shows the individual steps in converting keys in the example above. GUID1 references the InfoPackage. When you copy this to the D version, GUID1 is converted to GUID2. As in the example given in InfoPackage [Page 464], GUID2 is copied to GUID3 and to GUID4 for different customer source systems.



Example

The following example shows what you need to bear in mind when activating process chains using customer content.

All the source systems that you have selected as default systems are included when you activate a process chain in the customer system. For example, if you selected one SAP source system with release 4.0B and one with release 4.5B in the *Select Source Systems as Default* dialog box, the object collector collects one InfoPackage for each source system. If you activate a process chain with this same setting, both InfoPackages are transferred to the process chain:



We recommend that you only activate process chains for **one** source system as otherwise the subsequent process steps may be executed more than once (if the process chain is a periodic process chain) or too early (after the first InfoPackage is processed).

1. Scenario: Customer System with One Source System

Choose the source system before you activate the content.

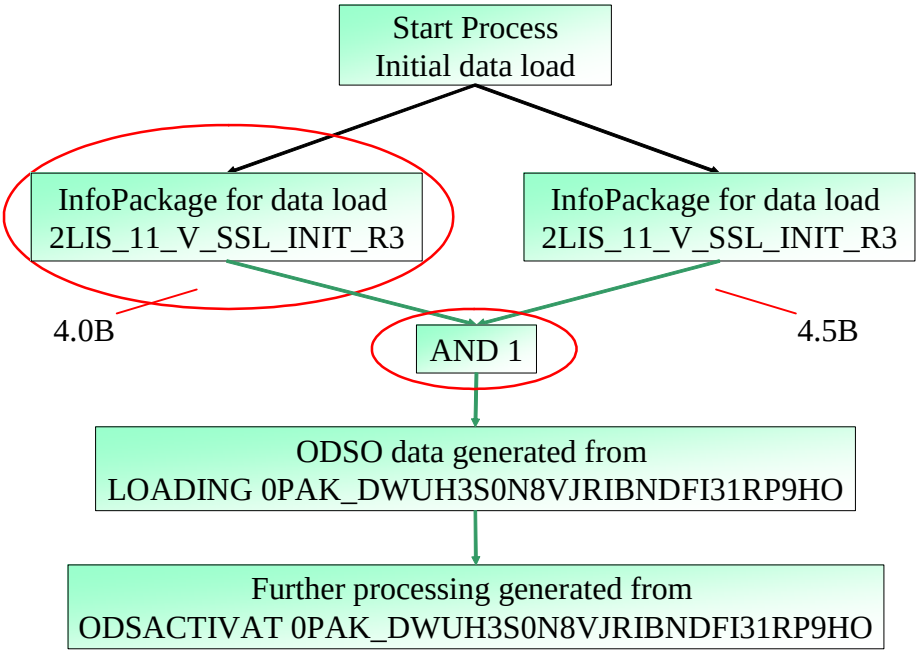
2. Scenario: Customer System with Several Source Systems

This involves additional manual efforts for the customer. They have to re-enter the connecting line in process chain maintenance after the event.

Variant 1

1. Select the relevant source system before you install the content.
2. Collect the InfoPackage and activate it with the selected source systems.
3. Choose **one** of the source systems before installing the content.
4. Collect the process chains and activate them with the selected source system.
5. Call the process chain maintenance. Add the InfoPackage process type “Load Data” to the process chain for the source system that you did **not** select before you activated the process chain. Add the process type “AND (Last)“ and include the necessary relations so that the new InfoPackages are included.

The following graphic shows how an InfoPackage is added subsequently for the source system in release 4.0B:



Variant 2

1. Select the relevant source system before you install the content.
2. Collect the process chains and activate them with the selected source systems.
3. Call the process chain maintenance. Delete the connecting lines that come from the “Data Load“ process. Add the process type “AND (Last)“ and include the necessary connecting lines so that the new InfoPackages are included.

Content Development System

Avoid changing the source system release after delivering your customer content. This can cause errors in a customer system with just one source system. The D version in the customer system is replaced when a new delivery takes place; however, the old active version is retained when you activate because the source system is not the same.

See also:

Transport and Delivery of Source-System Dependent Objects [Page 459]

Transfer Rule (SAP BW 3.x) [Page 466]

Transfer Structure (SAP BW 3.x) [Page 468]

InfoPackage [Page 464]

4.1.13.1.7.8 Error During Activation of Customer Content

When delivering source-system dependent objects within customer content, errors may occur which require a specially adapted error analysis due to the features of these objects.

Difficulties in Collecting Objects

Transport objects for transfer rules (SHMP) and InfoPackages (SHIP) are shadow delivery objects. If you activate the delivered customer content in the *BI Content* function area of the *Data Warehousing Workbench,* the system searches in the D versions as if shadow tables were not delivered. This is possible because after-import methods copied the shadow tables into the D version tables.

If difficulties arise when collecting objects, execute the following checks:

<table>
<tr><td colspan="3">Does the D version exist in the customer system?</td></tr>
<tr><td>Yes.

The problem lies in the object itself.</td><td colspan="2">No.

Does the object appear in the shadow table?</td></tr>
<tr><td></td><td>Yes.

The after-import method has not been executed successfully.

Copy again. You can use special function modules to do so.

For transfer rules:
RSAOS_PSEUDO_DVERSION_UPD writes a D version from the shadow tables and deletes the obsolete D version.

For InfoPackages:
RSSM_SHIP_PSEUDO_DVERSION_WRI creates the D version for an InfoPackage.</td><td>No.

A delivery error occurred.

Check whether the shadow table and the object exist in the content delivery and development system and in the transport requests.</td></tr>
</table>

See also:

Transfer Rule (SAP BW 3.x) [Page 466]

InfoPackage [Page 464]

4.1.13.1.8 Features when Transporting Roles

Delivery

Roles are not BI-specific objects and therefore do not exist in different versions (A/D). Instead, the delivery version and the active customer version are identified by namespaces. The delivering party determines a namespace in which they develop their BI Content.

BI Content roles are identified by the prefix SAP_BW_ and are copied to the SAP_BWC_ namespace when you activate BI Content (see Transporting Roles [External]).

If delivering BI customers use a registered namespace (such as /ABC/) (see Using Namespaces for Developing BI Objects [Page 450]), they **must** also use this namespace for the roles that they deliver. The system recognizes roles in this namespace as content roles. These roles are copied to a customer namespace when they are activated from the content.

This customer namespace is proposed by the delivering party but can be changed by the customer.

Activities

You can make various settings in *Content System Administration* (transaction RSOCONTENT) depending on whether you are working in a content system or a customer system. Choose *Namespace for Roles*. The *Customer Namespaces for BI Content* screen appears.

In a **Content system** you can create new delivery namespaces. These are automatically written to transport requests. You propose a customer namespace with the first delivery. Changes to the customer namespace are not effective if the delivery namespace has already been transported to the customer. The set customer namespace remains the active one for your customer.

In a **customer system**, you can change the customer namespaces for roles. We recommend that changes are only made at the start of a project if roles have not yet been transferred from the content.

If you change a customer namespace after roles have been transferred from content, the relations between the transferred roles and the roles to be delivered are lost. A second customer version is generated if you transfer a new role of this type from the content.

4.1.13.1.9 InfoAreas, Application Components, and InfoPackage Groups

Development and Delivery of Customer Content

1. Scenario: Nodes as Subnodes of the Root of the Tree on Required Position

In the content development system, in the *Modeling* functional area of the Data Warehousing Workbench, create one or more nodes as subnodes for your customer content object by using the context menu for the root node of the relevant tree. You can create as many additional nodes under this node as required.

For more information, see the InfoAreas, application components, and InfoPackage groups [Page 505] of the customer content.

2. Scenario: Nodes as Subnodes of the Root of the Tree on Required Position

If you create a node in the Data Warehousing Workbench tree for your customer content object, the system also registers nodes which mark their position in the hierarchy at the same time.

Details

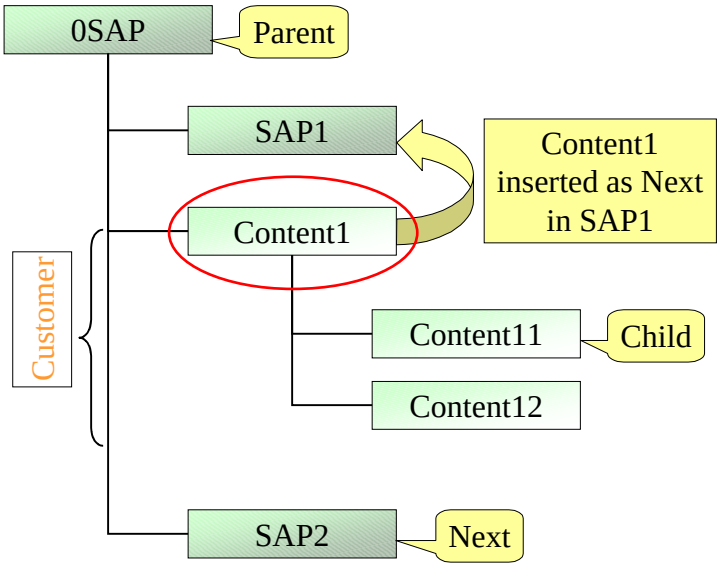
Name	Description	Field Name in the RSAPPL Table
Parent	Superordinate application component (parent node)	APPLNM_P

Child	Subordinate application component (child node) The system (only) enters the first subnode, even if there are several nodes.	APPLNM_C
Next	Next application component (neighbor node)	APPLNM_N

If you insert a node, under which your customer content objects hang, in a particular position on the child level under the root node of the respective tree, the system inserts your customer content node as NEXT in the predecessor node.

With the delivery of customer content, the system attempts to insert a customer content node in accordance with the table entries. If the tree in the customer system is structured differently than in the customer content development test system, and as a result, the node cannot be assigned, the system then decides to insert the node under *Non Assigned Nodes.*

The following graphic illustrates the relations between different objects for this node, under which all customer content objects are placed, using the example of the *Content1* node. *Content1* is to be inserted after the *SAPI* node:



To enable the system to insert the *Content1* node in the APPLNM_N field of its predecessor node *SAP1* in the RSAPPL table, you first need to set *SAP1* to **changeable**. Do this by assigning the value 2 to the CONTENT_CHANGE field in the RSOCONTCHNGABLE table: *External Objects Also Changeable in D Version- CTO Connection.* In this way, you ensure that the table entry (*Content 1* as NEXT in SAP1) is also delivered to all junior systems. For more information, see Object Changeability in Customer Content Systems [Page 453].

4.1.13.1.10 Comparing ABAP Code

Routines in Transfer and Update Rules

BI does **not** allow code to be compared in routines.

If you have to manage extensive ABAP code, it may be useful to store this code in a function module. Code in function modules can be compared using the AAK. If you are interested in this SAP product, contact SAP.

See also:

Start Routines in Transfer Rules [Page 283]

Creating Transfer Routines [Page 283]

Update Routines and Start Routines [Page 314]

Creating Update Routines [Page 317]

4.1.13.2 Customer Content in OLTP Systems

Purpose

Customers of SAP NetWeaver Business Intelligence (BI) can develop their own content also in SAP source systems (referred to as OLTP systems in the following text) and deliver it to their business units. In the following text, we refer to these business units as BI *customers*.

Integration

If you want to deliver customer content, you have to develop customer content in the OLTP system along with customer content in a BI system. For more information about how to develop content in a BI system, see Customer Content [Page 444] and Partner Content [Page 503]. The following sections are relevant for developing customer content in the OLTP system:

System Landscape [Page 445]

Using Namespaces for Developing BI Objects [Page 450]

Constraints

If you want to deliver customer content in the OLTP system, you have to develop your own customer content. You cannot change the delivered BI Content from SAP.

4.1.13.2.1 System Settings in the OLTP System

Purpose

You are able to define an OLTP system as a content development system using particular system settings. You can also create system requirements for generating customer content.

Process

1. Define your OLTP system as a content development system.

   To do this, select the *Content Development* field in *Content Settings Maintenance* (transaction RSA0) and save these settings.

2. To develop an application component hierarchy, specify the desired identification (/ABC/APCO for example) in the *Application Components - Hierarchies* field in *Content Settings Maintenance*. The name specified here acts as the default pre-setting for the system.

3. Create a development class or a package with a name space prefix (/ABC/PACK, for example). You then assign the objects that you create to this development class or package.

4. Ensure that the system type is defined as `CUSTOMER` in the global system parameters. You are able to check this by executing the function module **`TR_SYS_PARAMS`** in the function builder (*Tools* → *ABAP Workbench* → *Development*).

4.1.13.2.2 BI Content Delivery in OLTP Systems by SAP

Purpose

The standard procedure used in the delivery of BI Content by SAP is the model for the delivery of content by customers and partners. The basis of this process is also object versioning in the OLTP system.

The **DataSources** delivered by SAP with BI Content are in the delivery version (D version). If you want to transfer data from a OLTP system to a BI system using a BI Content DataSource, you have to first transfer this DataSource from the D version to the active version (A version).

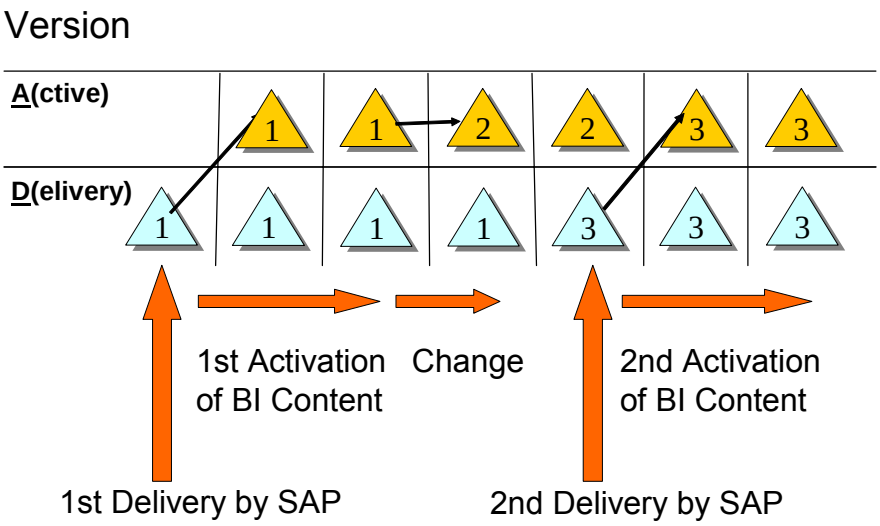
DataSources are assigned to application components in the OLTP system. You also need to transfer the **application component hierarchies** from the D version to the A version of the merged application component hierarchy (APCO_MERGED).This enables you to display the DataSources in this hierarchy in the OLTP system and in the source system view of the Data Warehousing Workbench - Modeling.

Upgrading BI Content has **no** effect on objects that you already use productively, since only the D version is overwritten.

If you activate the application component hierarchy and DataSources for BI Content, the active customer version is overwritten when you install the D version.

The extract structure (object type **R3TR TABL**) and extractor (object type: **LIMU FUNC**) behave differently. These objects do not have a D and A version. For this reason, we recommend a versioning with content generation. This prevents incompatible changes to the productive scenario from arising.

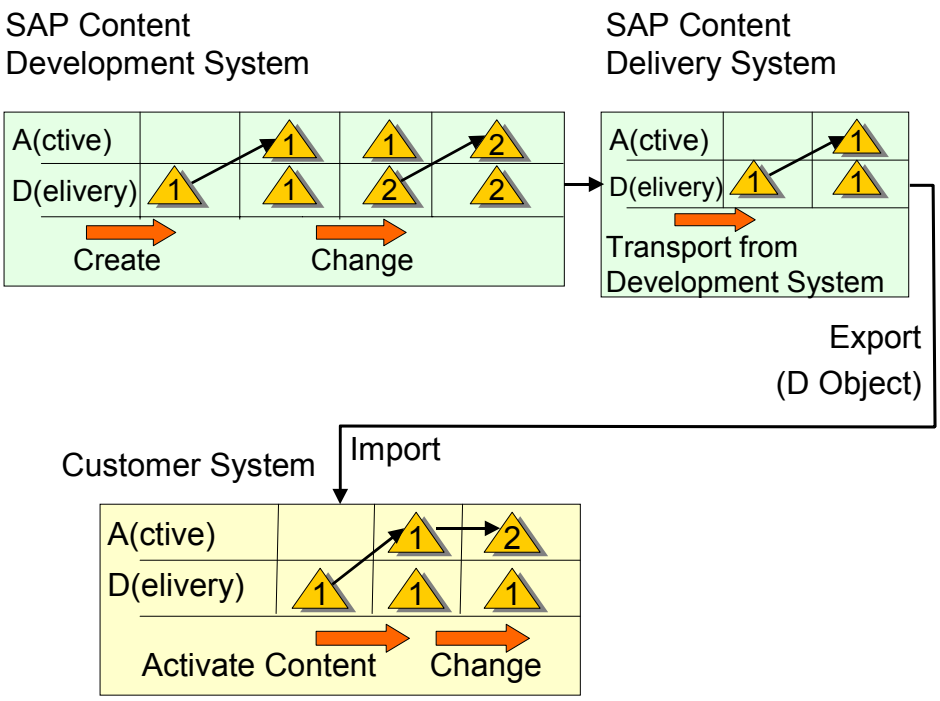
The following graphic shows how BI Content objects (DataSources and application component hierarchies) are used in a customer system with double delivery:



Process Flow

In the SAP content delivery system, the A and D version of DataSources and application component hierarchies are stored in identical forms. There are two transport objects for each object: an A and a D object. In accordance with the system settings, the system writes the suitable object to the transport request when transporting. When a content development system is involved, the system chooses the D object.

The next graphic illustrates how BI Content (DataSources and application component hierarchies) is delivered by SAP:



For objects which are seen as content objects in a system (with a changeable namespace and where the system is a content development system), the system automatically activates the DataSource in an after-import method for the D version.
For more information about the activation of BI Content in customer OLTP systems, see Installing Application Component Hierarchies [Page 540] and Installing BI Content DataSources [Page 540]. In this customer system, BI object changes are saved in the active version. The D version remains unchanged.

4.1.13.2.3 Creating Partner Content in the OLTP System

Purpose

Creating DataSources and an application component hierarchy in your own namespace enables you to deliver Business Content in the OLTP system.

Prerequisites

- You have applied for the namespace for the development and generation of objects, released it in the OLTP system and set it to changeable. You can find additional information here: Using a namespace for the development of BW Objects [Page 450].
- You have already made system settings in the OLTP system [Page 477].

Process

Creating a Partner DataSource

You can create a generic DataSource as a partner DataSource. You can find information about this procedure here: Maintaining Generic DataSources [Page 541].

When creating a DataSource in a Content development system, the system creates an A and D version. The DataSource is stored in identical form in both the A and D versions. There is only one version for the extract structure of the DataSource.

When you generate a DataSource, and in turn an extract structure, the system writes the following transport objects to a transport request:

Transport object	Description
R3TR OSOD /<Namespace>/<DataSource> /ABC/DATASOURCE1	D-version of DataSource
R3TR TABL /<Namespace>/ES<DataSource><four-character counter> or **R3TR TABL /ABC/<freely-chosen name of extract structure>** /ABC/DATASOURCE1	Extract structure

The active version of the DataSource (object type: OSOA) is not linked with the transport system and is not delivered.

SAP supports the creation of partner DataSources, which extract using database tables and views or using a function module.

When extracting using a view/table, note the following:

When extracting using a view/table, the system generates a proposal for the extract structure name when you enter a name for the DataSource: **/<Namespace>/ES<DataSource><four-character pointer>**, z. B. /ABC/ESDATASOURCE10001.

You can change the name of the extract structure.

This is important when you want to make incompatible changes to the extract structure (such as changing a field type) and the productive scenario is to remain in place in the delivered system. In the screen to create a DataSource, enter a new name for the extract structure. After generating the changed DataSource, the extract structure is regenerated under this name.

When extracting using a function module, note the following:

In this case, the system does **not** generate a proposal for the extract structure. Enter the name of the extract structure that you have previously created in your namespace in the ABAP Dictionary.

If you choose to make incompatible changes to the extract structure (such as changing a field type), create an extract structure with a new name in the ABAP Dictionary (as a modified copy of the old structure). In this way, you also ensure that the productive scenario remains in place in the delivered system and that the extract structure is not overwritten.

Creating a Partner Application Component Hierarchy

1. Enter transaction code rsa8 in the command field.

   The *Selection of a Hierarchy* dialog box appears.

2. Enter the name of the hierarchy if not already proposed.
3. Choose *Create.*

4. Choose Create. Other information about editing application component hierarchies can be found under Editing DataSources and Application Component Hierarchies [Page 550].

   After saving the hierarchy, the system writes the transport object **R3TR DSAD /ABC/<Application component hierarchy>** to a transport request.

   Upon saving the system creates a D version and an identical A version for the application component hierarchy, which can only be transported in its entirety, as was the case for DataSources.

   If you only want to deliver individual DataSources, you do not need to have your own application component hierarchy. Check whether an existing application component is suitable for the grouping and delivery of your DataSources. DataSources that have not been assigned to an application component appear in the application component hierarchy under the node NODESNOTCONNECTED. They can be edited and used in the standard way.

   If, when changing a system for example, you want to keep the D version of a hierarchy, you need to execute report **RSA1APDC** before editing the hierarchy. This report generates an A version from the imported D version.

Result

The D versions of the generated DataSources and application component hierarchies, as well as the extract structures and, where applicable, function modules, appear in a transport request. These can be transported into the Content test or –delivery system and from here delivered to partners or business areas.

4.1.13.2.4 Delivery and Use of Customer Content in OLTP Systems

Purpose

You can use this process to enable your business areas or other customers to import/use customer content. The basis of the process is object versioning for customer content as well (see BI Content Delivery in OLTP Systems by SAP [Page 478]).

Prerequisites

Prerequisites in the development and test/delivery system for customer content:

You have already made system settings in the OLTP system [Page 477].

Prerequisites in the customer system into which content is imported:

- You have ensured that the OLTP system in which the delivered content is to be delivered, is **not** defined as a content development system. You can check this in the *Content Development Maintenance* screen (transaction RSA0).
- You have created the namespace as changeable in the OLTP system (role **C**).
- You have ensured that a generation namespace has been released in the BI system in which the DataSources are to be replicated.
- You have created and released the same generation namespace in the OLTP system.

See also: Using Namespaces for Developing BI Objects [Page 450]

Process Flow

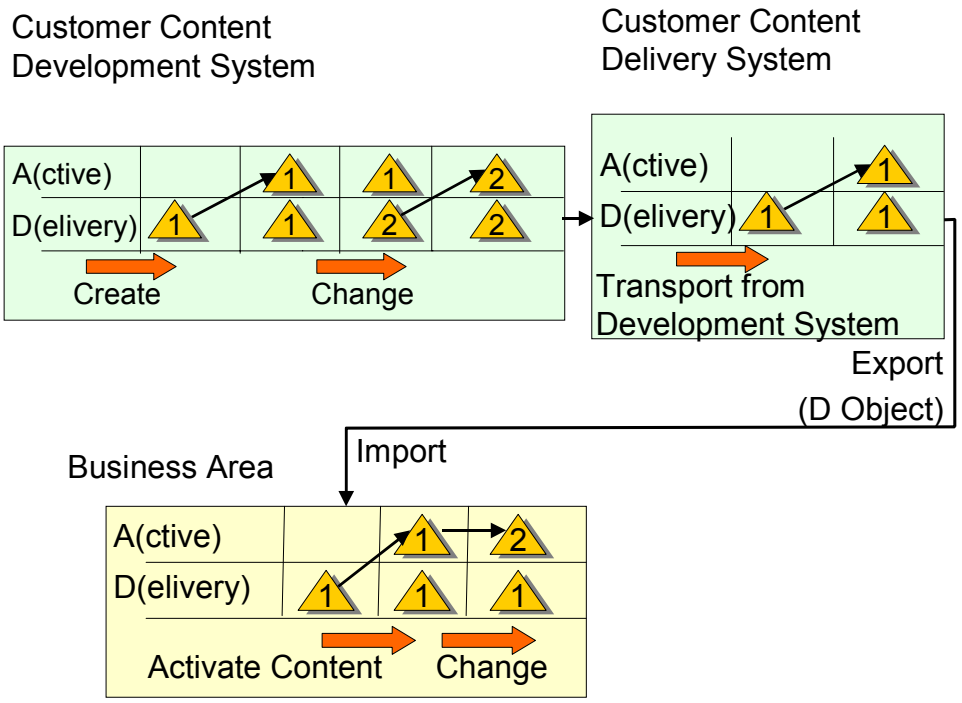
Deliver the objects (DataSources and application component hierarchies in the D version, extract structures, extractors) from the content test system or delivery system using a delivery transport (see System Landscape [Page 445]). Note that for objects in A and D versions, the system only writes the D versions of the object to the transport request. You ensure this by selecting the *Content Development* field in the *Content Development Maintenance* screen (transaction RSA0) in the system.

1. After the import of the D versions, your customers (business area) can activate the DataSources and application component hierarchies and use them similarly to the BI Content delivered by SAP.

   Using application component hierarchies delivered as customer content:

   All application component hierarchies in the D version are merged in a hierarchy upon activation in the OLTP system (transport object **R3TR DSAA APCO_MERGED**). This hierarchy also contains the customer content hierarchy. The merged hierarchy is replicated as a **single** object in BI. The customer application component hierarchy is therefore not displayed as a separate object in the BI system.

The following graphic illustrates the process flow for DataSources and application component hierarchies:



For objects which are seen as content objects in a system (with a changeable namespace and where the system is a content development system), the system automatically activates the DataSource in an after-import method for the D version.
For more information about the activation of BI Content in OLTP systems in business areas, see Installing BI Content DataSources [Page 540] and Installing Application Component Hierarchies [Page 540]. In this system, object changes are saved in the active version. The D version remains unchanged.

4.1.13.2.5 unctions in the SAP Source System

Use

The BI Service API (SAPI) is a technology package in the SAP source system that enables the close integration of data transfer from SAP source systems into a BI system.

The SAPI allows you to

- make SAP application extractors available as a basis for data transfer into BI
- carry out generic data extraction
- use intelligent delta processes
- access data in the source system directly from BI (VirtualProvider support)

With transaction SBIW, the SAPI provides an implementation guide in the SAP source system that includes the activities necessary for data extraction and data transfer from an SAP source system into BI.

Irrespective of the type of SAP source system, Customizing for extractors comprises activities that belong to the scope of SAPI:

- general settings for data transfer from a source system into BI
- the option of installing BI Content delivered by SAP
- the option of maintaining generic DataSources
- the option of postprocessing the application component hierarchy and DataSources on a source system level

In addition to the activities that are part of the scope of SAPI, Customizing for extractors for OLTP and further SAP source systems may contain source-system specific settings for application-specific DataSources.

Features

General Settings

General settings include the following activities:

- Maintaining control parameters for data transfer [Page 536]
- Restricting authorizations for extraction [Page 538]
- Monitoring the delta queue [Page 538]

Installing BI Content Delivered by SAP

DataSources delivered with the BI Content by SAP and those delivered by partners appear in a delivery version (D version). If you want to use a partner or BI Content DataSource to transfer data from a source system into BI, you need to transfer this DataSource from the D into the active (A) version.

In the source system, the DataSources are assigned to specific application components. If you want to display the DataSources in BI in the DataSource tree of the Data Warehousing Workbench according to this application component hierarchy, you need to transfer them from the D version into the A version in the source system.

Transferring data from an OLTP system or other SAP source systems

Note: You need to make settings for some BI Content DataSources before you can transfer data into BI. These settings are listed in transaction SBIW in the *Settings for Application-Specific DataSources* section. You can only find this section in those SAP source systems for which it is relevant.

The following activities are associated with installing BI Content:

- Transferring application component hierarchies [Page 540]
- Installing Business Content DataSources [Page 540]

Generic DataSources

Regardless of the specific application, you can use generic data extraction to extract data from any transparent tables, database views or SAP Query functional areas. You do not need to program in ABAP. You can also use function modules for generic data extraction.

In this way, you can use your own DataSources for transaction data, master data attributes or texts. The data for such DataSources is read generically and then transferred into BI.

Generic DataSources allow you to extract data which cannot be supplied to BI either with the DataSources delivered with BI Content or with customer-defined DataSources of the application.

For more information, see Maintaining Generic DataSources [Page 541].

Postprocessing DataSources

You can adapt existing DataSources to suit your requirements as well as edit the application component hierarchy for the DataSources.

For more information, see Editing DataSources and Application Component Hierarchies [Page 550].

4.1.13.2.5.1 Maintaining Control Parameters for Data Transfer

Procedure

Maintain entries for the following fields:

1. Source system

   Enter the logical system for your source client and assign a control parameter to it.

   For information about source clients, see the source system under *Tools* → *Administration* → *Management* → *Client Management* → *Client Maintenance*.

2. Maximum size of the data package

   When you transfer data into BI, the individual data records are sent to BI in packages of variable size. You use this parameter to control the typical size of a data package of this type.

   If you do not maintain an entry, the data is transferred with the default setting of 10,000 kBytes per data package. However, the required memory depends not only on the data

package size setting, but also on the width of the transfer structure, the required memory of the affected extractor, and, for large data packages, the number of data records in the package.

3. Maximum number of rows in a data package

   For large data packages, the required memory mainly depends on the number of data records that are transferred with the package. You use this parameter to control the maximum number of data records that you want the data package to contain.

   By default, the system transfers a maximum of 100,000 records per data package.

   The maximum main memory required per data package is approximately 2 X 'Max. Rows' X 1000 bytes.

4. Frequency

   By specifying a frequency you determine the number of data IDocs after which an info IDoc is to be sent. In other words, how many data IDocs are described by a single info IDoc.

   The frequency is set to 1 by default. This means that an info IDoc follows after each data IDoc. You should choose a frequency between 5 and 10, but not greater than 20.

   The larger the package size of a data IDoc, the lower you should set the frequency. As a result, you get information about the data load status during the data upload at relatively short intervals.

   In the BI monitor, you can see from each info IDoc whether the load process was successful. If this is the case for all data IDocs described in an info IDoc, the traffic light in the monitor is green. Info IDocs contain information about whether the data IDocs were correctly uploaded.

5. Maximum number of parallel processes for the data transfer

   An entry in this field is only required as of Release 3.1I.

   Enter a value greater than 0. The maximum number of parallel processes is set to 2 by default. The optimal choice of the parameter depends on the configuration of the application server that you are using for the data transfer.

6. Target system of a batch job

   Enter the name of the application server on which you want to process the extraction job.

   To get the name of the application server, choose *Tools → Administration → Monitor → System Monitoring → Server*. The *Host* column displays the name of the application server.

7. Maximum number of data packages in a delta request

   You use this parameter to set the maximum number of data packages in a delta request or in the repeat of a delta request (repair).

   Only use this parameter if you are expecting delta requests with a very large volume of data. In this case, you allow more than 1000 data packages to be generated in a request, while retaining an appropriate data package size.

   As before, there are no limits for initials values or the value 0. A limit is only applied if you have a value that is greater than 0. However, for consistency reasons this number is not always strictly adhered to. Depending on the extent to which the data in the qRFC queue is compressed, the actual limit can deviate by up to 100 from the specified value.

4.1.13.2.5.2 Restricting Authorizations for Extraction

Use

You use this function to exclude DataSources from the extraction. Data that is stored in these DataSources is not transferred into BI.

Use this function to exclude DataSources from the extraction for individual BI systems. If you want to exclude a DataSource from the extraction for all connected BI systems, in the post-processing of DataSources, choose **editing DataSources and application component hierarchies [Page 550]** and delete the DataSource.

Procedure

1. Choose *New Entries*.
2. Choose the DataSource that you want to exclude from the extraction.
3. Choose the BI system into which you no longer want data from this DataSource to be extracted.
4. In the *Extr. Off* field, specify that the DataSource is to be excluded from the extraction.
5. Save your entries and specify a transport request.

4.1.13.2.5.3 Delta Queue Check

Use

The delta queue is a data store in the source system into which data records are written automatically. The data records are written to the delta queue either using an update process in the source system (for example with FI documents) or are extracted using a function module when data is requested from BI (for example, LIS extraction prior to BW 2.0).

With a delta request, the data records are transferred into BI from the scheduler.

The data is stored in compressed form in the delta queue. It can be requested from several BI systems. The delta queue is also repeat enabled; it stores the data from the last extraction process. The repeat mode of the delta queue is specific to the target system.

If the extraction structure of a DataSource is changed after data is written into the delta queue but before the queue data is read (for example, when you upgrade), you can tell which structure in the delta queue the data was written to from the data itself. The queue monitor contains fields that were not filled before but are now filled and/or fields that were filled before but are no longer filled.

You use this function to check the delta queue.

Features

The status symbol shows whether an update into a delta queue is activated for a particular DataSource. The delta queue is active if the status symbol is green; it is filled with data records when there is an update process or data request from BI. The delta method has to be initialized successfully in the scheduler in BI before a delta update can take place.

You can carry out the following activities:

- Display data records
- Display the current status of the delta-relevant field
- Refresh
- Delete the queue
- Delete queue data

Activities

Displaying Data Records

1. To check the amount and type of data in the delta queue, select the delta queue and choose *Display Data Records.*
2. A dialog box appears in which you can specify how you want to display the data records.
    a. You can select the data packages that contain the data records you want to see.
    b. You can display specific data records in the data package.
    c. You can simulate the extraction parameters to select how you want to display the data records.
3. To display the data records, choose *Execute.*

Displaying Current Status of Delta-Relevant Field

For DataSources that support generic deltas, you can display the current value of the delta-relevant field in the delta queue. In the *Status* column, choose *Detail*. The value displayed includes the largest value for the last extraction with reference to the delta-relevant field. It is the lower limit for the next extraction.

Refreshing

If you select refresh,

- newly activated delta queues are displayed
- new data records that have been written to the delta queue are displayed
- data records that have been deleted by the time the system reads the data records are not displayed

Deleting Queue Data

To delete the data in a delta queue for a DataSource, select the delta queue and in the context menu, choose *Delete Data.*

If you delete data from the delta queue, you do not have to reinitialize the delta method to write the DataSource data records into the delta queue.

Note that data is also deleted that has not yet been read from the delta queue. As a result, any existing delta update is invalidated. Only use this function when you are sure that you want to delete all queue data.

Deleting Queues

You can delete the entire queue by choosing *Queue* → *Delete Queue*. You need to reinitialize the delta method before you can write data records for the related DataSource into the delta queue.

4.1.13.2.5.4 Installing Application Component Hierarchies

You use this function to install and activate application component hierarchies delivered by SAP or by partners.

After the DataSources are replicated in BI, this application component hierarchy is displayed with the transferred DataSources in the source system view of the Data Warehousing Workbench – Modeling. In BI, choose the DataSource overview from the context menu (right-mouse click) for the source system.

If you activate the BI Content application component hierarchy, the active customer version is overwritten when you install the BI Content version.

For information about changing the installed application component hierarchy, see Editing DataSources and Application Component Hierarchies [Page 550].

4.1.13.2.5.5 Installing BI Content DataSources

Use

You use this function to transfer and activate DataSources delivered with BI Content and, where applicable, partner DataSources delivered in their own namespaces. After installing BI Content DataSources you can extract data from all the active DataSources that you have replicated in BI and transfer this data to all connected BI systems.

Activities

The *Install DataSources from BI Content* screen displays the DataSources in an overview tree. This tree is structured in accordance with the application components assigned to you.

1. In the application component hierarchy, select the nodes for which you want to install DataSources in the active version. To do this, position the cursor on the node and choose *Highlight Subtree.*

   The DataSources and subtrees below the node are selected.

2. Choose *Select Delta*.

   DataSources where the system found differences between the active and the delivered version (due to changes to the extractor, for example) are highlighted in yellow.

3. To analyze the differences between active and delivered versions of a particular DataSource, select the DataSource and choose *Version Comparison.* The application log contains further information about the version comparison.

4. To transfer a DataSource from the delivery version to the active version, select it in the overview tree by choosing *Highlight Subtree* and choose *Transfer DataSources.*

   If an error occurs, the error log appears.

   Regardless of whether data has been successfully transferred into the active version, you can call the log by choosing *Display Log.*

With a metadata upload (when you replicate DataSources in BI), the active version of the DataSource is made known to BI.

When you activate BI Content DataSources, the system overwrites the active customer version with the SAP version.

You can only search for DataSources or other nodes in expanded nodes.

For information about changing the installed DataSources, see Editing DataSources and Application Components [Page 550].

4.1.13.2.5.6 Maintaining Generic DataSources

Use

Regardless of the application, you can create and maintain generic DataSources for transaction data, master data attributes or texts from any transparent table, database view or SAP Query InfoSet, or using a function module. This allows you to extract data generically.

Procedure

Creating Generic DataSources

1. Select the DataSource type and specify a technical name.
2. Choose *Create.*

   The screen for creating a generic DataSource appears.
3. Choose the application component to which you want to assign the DataSource.
4. Enter the descriptive texts. You can choose any text.
5. Select the datasets from which you want to fill the generic DataSource.
   a. Choose *Extraction from View* if you want to extract data from a transparent table or a database view. Enter the name of the table or the database view.

      After you generate the DataSource, you have a DataSource with an extraction structure that corresponds to the database view or transparent table.

      For more information about creating and maintaining database views and tables, see the *ABAP Dictionary Documentation*.

   b. Choose *Extraction from Query* if you want to use a SAP Query InfoSet as the data source. Select the required InfoSet from the InfoSet catalog.

      Notes on Extraction Using SAP Query [Page 549]

      After you generate the DataSource, you have a DataSource with an extraction structure that corresponds to the InfoSet.

      For more information about maintaining the InfoSet, see the *System Administration* documentation.

   c. Choose *Extraction Using FM* if you want to extract data using a function module. Enter the function module and extraction structure.

The data must be transferred by the function module in an interface table E_T_DATA.

Interface Description and Extraction Process [Page 546]

For information about the function library, see the *ABAP Workbench: Tools* documentation.

d. With texts you also have the option of extracting from fixed values for domains.

6. Maintain the settings for delta transfer, as required.
7. Choose *Save*.

When performing extraction, note SAP Query: Assigning to a User Group [Page 549].

Note when extracting from a transparent table or view:

If the extraction structure contains a key figure field that references a unit of measure or a currency unit field, this unit field has to be included in the same extraction structure as the key figure field.

A screen appears on which you can edit the fields of the extraction structure.

8. Edit the DataSource:

- Selection

  When you schedule a data request in the BI scheduler, you can enter the selection criteria for the data transfer. For example, you can determine that data requests are only to apply to data from the previous month.

  If you set the *Selection* indicator for a field within the extraction structure, the data for this field is transferred in correspondence with the selection criteria in the scheduler.

- Hide field

  You set this indicator to exclude an extraction structure field from the data transfer. The field is no longer available in BI when you set the transfer rules or generate the transfer structure.

- Inversion

  Reverse postings are possible for customer-defined key figures. Therefore inversion is only active for certain transaction data DataSources. These include DataSources that have a field that is marked as an inversion field, for example, the **update mode** field in DataSource **0FI_AP_3**. If this field has a value, the data records are interpreted as reverse records in BI.

  If you want to carry out a reverse posting for a customer-defined field (key figure), set the *Inversion* indicator. The value of the key figure is transferred to BI in inverted form (multiplied by –1).

- Field only known in exit

  You can enhance data by extending the extraction structure for a DataSource by adding fields in append structures.

  The *Field Only Known in Exit* indicator is set for the fields of an append structure; by default these fields are not passed to the extractor from the field list and selection table.

  Deselect the *Field Only Known in Exit* indicator to enable the Service API to pass on the append structure field to the extractor together with the fields of the delivered extract structures in the field list and in the selection table.

9. Choose *DataSource → Generate.*

The DataSource is saved in the source system.

Maintaining Generic DataSources

- Change DataSource

  To change a generic DataSource, in the initial screen of DataSource maintenance, enter the name of the DataSource and choose *Change.*

  You can change the assignment of a DataSource to an application component or change the texts of a DataSource. Double-click on the name of the table, view, InfoSet or extraction structure to get to the appropriate maintenance screen. Here you make the changes to add new fields. You can also completely swap transparent tables and database views, though this is not possible with InfoSets. Return to DataSource maintenance and choose *Create*. The screen for editing a DataSource appears. To save the DataSource in the SAP source system, choose *DataSource → Generate.*

  If you want to test extraction in the source system independently of a BI system, choose *DataSource → Test Extraction.*

- Delta DataSource

  On the *Change Generic DataSource* screen, you can delete any DataSources that are no longer relevant. If you are extracting data from an InfoSet, delete the corresponding query. If you want to delete a DataSource, make sure it is not connected to a BI system.

For more information about extraction using SAP Query, see Extraction Using SAP Query [Page 548].

4.1.13.2.5.6.1 Delta Transfer to BI

The following update modes are available in BI:

- Full update

  A full update requests all data that meets the selection criteria you set in the scheduler.

- Delta update

  A delta update only requests data that has appeared in the source system since the last load.

- Initializing the delta process

  You need to initialize a delta process before you can execute it. The initialization selections are copied to load the delta records.

With large volumes of data, you can only ensure a performance-optimized extraction from the source system if you use a delta process.

In the maintenance of the generic DataSource, you can set up a delta for master data attributes and texts. You can also set up a generic delta using a (delta-relevant) field with a monotonically increasing value.

Setting Up an ALE Delta for Master Data Attributes or Texts

Master data attributes or texts for which you want to use a delta transfer have to fulfill two prerequisites:

1. Data must be extracted generically using a transparent table or database view.

2. A change document object must be available that can update the complete key of the table (or view) used for data extraction in combination with one of the tables on which the change document object is based.

The required control entries are delivered for the most important master data attributes and texts. By including a maintenance interface for control entries in the maintenance of generic DataSources or InfoSources, you can use the delta transfer for other master data attributes or texts.

To generate the control entry for master data attributes or texts that is required for BI, proceed as follows:

1. For an attribute or text DataSource, choose *DataSource* → *ALE Delta*.
2. Enter the table and the change document object that you want to use as a basis for the delta transfer.

   An input help for the *Table Name* field searches all possible tables for a suitable key.
3. Confirm your entries.

   If you have entered a usable combination of table and change document object, the extraction structure fields are listed in the table below. The status in the first column shows whether changing the master data in this field causes the system to transfer the delta record.
4. Save the settings to generate the required control entry.

   Delta transfer is now possible for master data and texts.

After the DataSource has been generated, you can see this on the *DataSource: Edit Customer Version* screen; the *Delta Update* field is selected.

You need two separate entries if you want to transfer delta records for texts and master data attributes.

Generic Delta

If a field exists in the extraction structure of a DataSource and the field contains values that increase monotonically over time, you can define delta capability for this DataSource. If a delta-relevant field of this type exists in the extraction structure (a timestamp, for example), the system determines the data volume transferred in the delta mode by comparing the maximum value transferred with the last load with the amount of data that has since entered the system. Only the new data is transferred.

To get the delta, generic delta management translates the update mode into a selection criterion. The selections of the request are enhanced with an interval for the delta-relevant field. The lower limit of the interval is taken from the previous extraction. The upper limit is taken from the current value, for example, the timestamp at the time of extraction. You use security intervals to ensure that all data is taken into account during extraction (see below). After the data request is transferred to the extractor and the data is extracted, the extractor informs generic delta management that the pointer can be set to the upper limit of the previously determined interval.

The delta for generic DataSources cannot be used with a BI system release prior to 3.0. In older SAP BW releases, the system does not replicate DataSources for master data and texts that were delta-enabled using the delta for generic DataSources.

Determining Generic Delta for a DataSource

1. Choose *Generic Delta*.
2. In the dialog box that appears, specify the delta-determining field and the type of this field.
3. Maintain the settings for the generic delta:
   a. Enter a security interval.

      The purpose of a security interval is to make the system extract records with the next extraction that appear during the extraction process but which have not yet been extracted (because, for example, they have not been saved).

      You can add a security interval to the upper limit or lower limit of the interval.

      You should only specify a security interval for the lower limit if the delta process produces a new status for the changed records (when the status is overwritten in BI). In this case, duplicate data records that may arise with a security interval of this type have no affect in BI.

   b. Choose the delta type for the data that you want to extract.

      You use the delta type to determine how the extracted data is interpreted in BI and the data targets to which it can be updated.

      With the *Additive Delta* delta type, the record to be loaded for cumulative key figures only returns the change to the respective key figure. The extracted data is added into BI. DataSources with this delta type can write data to DataStore objects and InfoCubes.

      With the *New Status for Changed Records* delta type, each record to be loaded returns the new status for all key figures and characteristics. The values in BI are overwritten. DataSources with this delta type can write data to DataStore objects and master data tables.

   c. Specify whether the DataSource supports real-time data acquisition.
4. Save your entries.

   Delta transfer is now possible for this DataSource.

After the DataSource has been generated, you can see this on the *DataSource: Edit Customer Version* screen; the *Delta Update* field is selected.

In systems as of Basis Release 4.0B, you can display the current value of the delta-relevant field in the delta queue.

Example of Determining Selection Intervals with Generic Delta

Security interval upper limit

The delta-relevant field is a timestamp.

The timestamp that was read last is 12:00:00. Delta extraction begins at 12:30:00. The security interval for the upper limit is 120 seconds. The selection interval for the delta request is: 12:00:00 to 12:28:00. When the extraction is finished, the pointer is set to 12:28:00.

Security interval lower limit

The delta-relevant field is a timestamp. After images are transferred. In BI the record is overwritten with the post-change status, for example, for master data. Any duplicate data records do not affect BI.

The last read timestamp is 12:28:00. Delta extraction begins at 13:00. The security interval for the lower limit is 180 seconds. The selection interval for the delta request is: 12:25:00 to 13:00:00. When the extraction is finished, the pointer is set to 13:00:00.

4.1.13.2.5.6.2 Function Module: Interface Description and Procedure

A description of the interface for a function module that is used for generic data extraction:

Importing Parameter

I_DSOURCE type SRSC_S_IF_SIMPLE-DSOURCE DataSource

I_INITFLAG type SRSC_S_IF_SIMPLE-INITFLAG Initialization call

I_MAXSIZE type SRSC_S_IF_SIMPLE-MAXSIZE package size

I_REQUNR type SRSC_S_IF_SIMPLE-REQUNR request number

Tables

I_T_SELECT type SRSC_S_IF_SIMPLE-T_SELECT

I_T_FIELDS type SRSC_S_IF_SIMPLE-T_SELECT

E_T_DATA

Exceptions

NO_MORE_DATA

ERROR_PASSED_TO_MESS_HANDLER

Details on Individual Parameters

- I_INITFLAG

  This parameter is set to ‘X’ when the function module is called up for the first time, then to ‘ ‘.

- I_MAXSIZE

  This parameter contains the number of lines expected within a read call.

Extraction Process

The function module is called up again and again during an extraction process:

1. Initialization call:

   Only the request parameters are transferred to the module here. The module is as yet unable to transfer data.

2. First read call:

   The extractor returns the data typified with the extract structure in an interface table. The number of rows the system expected is determined in the request parameter (I_MAXSIZE).

3. Second read call:

The extractor returns the data enclosed within the first data package in a separate package with I_MAXSIZE rows.

4. The system calls up the function module again and again until the module returns the exception NO_MORE_DATA. Data cannot be transferred in the call in which this exception is called up.

Example

An example of a function module that meets these demands is RSAX_BIW_GET_DATA_SIMPLE. A simple way of creating a syntactically correct module is to copy it into its own function group and then to cope the rows of the top-include of function group RSAX (LRSAXTOP) into the top-include of its own function group. Afterwards, the copied function module must be adjusted to the individual requests.

4.1.13.2.5.6.3 Testing Extraction

Use

You can use this function to test extraction from DataSources that were created using the maintenance for the generic DataSource. After the test extraction, you can display the extracted data and the associated logs.

Procedure

1. Choose **DataSource** → *Test Extraction*.

   A screen appears in which you can set parameters and selections for the test extraction.

2. Enter a request no. for the test extraction via a function module.
3. Enter how many data records are to be read with each extractor call.
4. The extractor is called up by the Service API until no more data is available. In the *Display Extr. Calls* field, you can specify the maximum number of times the extractor is to be called. This enables you to restrict the no. of data packages when testing the extraction. With a real extraction, the system transfers data packages until is no longer able to find any more data.
5. Depending on the definition of the DataSource, you can test the extraction in various update modes.

   For DataSources that support the delta method, you can also test deltas and repeats as well as the full update.

   The modes delta and repeat are only available for testing when the extractor supports a mode in which the system reads the data but does not modify the delta management status tables.

   - To avoid errors in BW, the timestamp or pointer that was set in delta management must not be changed during testing.
   - Before you are able to test the extraction in a delta mode in the source system, you need to have carried out an initialization of

the delta method or a simulation of such an initialization for this DataSource.

You can test the transfer of an opening balance for non-cumulative values.

6. Specify selections for the test extraction.

   Only those extract structures fields you have selected in DataSource maintenance can be selected.

   To enter several selections for a field, insert new rows for this field into the selection table.

7. Choose whether you want to execute the test extraction in debug mode or by tracing an authorization trace.

   If you test the extraction in the debug mode, a breakpoint is set just before the extractor initialization call.

   For information on the debugger, see **ABAP Workbench: Tools**.

   If you set an authorization trace, you can call it after the test by choosing *Display Trace.*

8. Start the extraction.

Result

If the extraction was successful, a message appears that specifies the number of extracted records. The buttons *Display List, Display Log* and *Display Trace* (optional) appear on the screen. You can use *Display List* to display the data packages. By double-clicking on the number of records for a data package, you get to a display of the data records. Choose *Display Log* to display the application log.

4.1.13.2.5.6.4 Extraction Using SAP Query

SAP Query is a comprehensive tool for defining reports. It uses many different forms of reporting. It allows users to define and execute their own evaluations of data in the SAP system without requiring ABAP programming know-how.

To define the structure of evaluations, you enter texts in SAP Query and select fields and options. InfoSets and functional groups allow you to easily select the relevant fields.

An InfoSet is a special view of a set of data (logical database, table join, table, sequential file). It serves as the data source for SAP Query. An InfoSet determines which tables or fields of these tables are referenced in an evaluation. InfoSets are usually based on logical databases.

The maintenance of InfoSets is one component of SAP Query. When an InfoSet is created, a data source is selected in an application system. Since a data source can have a large number of fields, fields can be combined into logical units; the functional groups. Functional groups are groups of several fields that form a logical unit within an InfoSet. Any fields that you want to use in an extraction structure have to be assigned to a functional group. In generic data extraction using an InfoSet, all the fields of all functional groups for this InfoSet are available.

The relevance of SAP Query to BI lies in the definition of the extraction structure by selecting fields of a logical database, a table join or other datasets in an InfoSet. This allows you to use generic data extraction for master or transaction data from any InfoSet. A query is generated for an InfoSet. The query gets the data and transfers it to the generic extractor.

InfoSets represent an additional, easily manageable data source for generic data extraction. They allow you to use logical databases from all SAP applications, table joins, and further

datasets as data sources for BI. For more information about SAP Query, and InfoSets in particular, see the SAP Query documentation -> System Administration.

In the following section, the terms SAP Query and InfoSet are used independently of the source system release. Depending on the source system release, SAP Query is the same as an ABAP Query or ABAP/4 query. The InfoSet is also called a functional area in some source system releases.

4.1.13.2.5.6.4.1 Notes on Extraction Using SAP Query

Client Dependency

InfoSets are only available if you have created them globally, independent of a client. You set this global area in the initial screen of InfoSet maintenance under *Environment → Work Areas.*

Size Limits When Extracting Data Using SAP Query InfoSets

If you are using an InfoSet to extract data, the system first collects all data in the main memory. The data is transferred to the BI system in packages using the Service API interface. The size of the main memory is therefore important with this type of extraction. It is suitable for limited datasets only.

As of SAP Web Application Server 6.10, you can extract mass data using certain InfoSets (tables or table joins).

See also:

Extraction Using SAP Query [Page 548]

4.1.13.2.5.6.4.2 SAP Query: Assignment to a User Group

If you want to extract your data from an InfoSet, the InfoSet must be assigned to a user group before the DataSource can be generated. This is necessary as the extraction is processed from an InfoSet using a query that comprises all fields of the InfoSet. In turn, this query can only be generated when the InfoSet is assigned to a user group.

Releases up to 3.1I

In releases up to 3.1I, a screen appears in which you have to specify a user group as well as a query name. The user group must be specified using the value help. In other words, it must already have been created. You can get more information about creating user groups in the SAP Query documentation, in the section *System Management → Functions for Managing User Groups*.

A separate query is required for an InfoSet each time it is used in a DataSource. For this reason, enter a query name that was previously not in the system.

The query is generated after you confirm your entries.

Releases from 4.0A

In releases as of 4.0A, the InfoSet for the extract structure of the new DataSource is automatically assigned to the pre-finished system user group. A query is automatically generated by the system.

4.1.13.2.5.7 Editing DataSources and Application Component Hierarchies

Use

To adapt existing DataSources to your requirements, you can edit them in this step before transporting them from a test system into a productive system.

In this step you can also postprocess the application component hierarchy.

Procedure

DataSource

Transporting DataSources

Select the DataSources that you want to transport from the test system into the productive system and choose *Transport*. Specify a development class and a transport request so that the DataSources can be transported.

Maintaining DataSources

To maintain a DataSource, select it and choose *Maintain DataSource*. The following editing options are available:

- Selection

  When you schedule a data request in the BI scheduler, you can enter the selection criteria for the data transfer. For example, you can determine that data requests are only to apply to data from the previous month.

  If you set the *Selection* indicator for a field within the extraction structure, the data for this field is transferred in correspondence with the selection criteria in the scheduler.

- Hide field

  You set this indicator to exclude an extraction structure field from the data transfer. The field is no longer available in BI when you set the transfer rules or generate the transfer structure.

- Inversion

  Reverse postings are possible for customer-defined key figures. Therefore inversion is only active for certain transaction data DataSources. These include DataSources that have a field that is marked as an inversion field, for example, the **update mode** field in DataSource **0FI_AP_3**. If this field has a value, the data records are interpreted as reverse records in BI.

  Set the *Inversion* indicator if you want to carry out a reverse posting for a customer-defined field (key figure). The value of the key figure is transferred to BI in inverted form (multiplied by –1).

- Field only known in exit

  You can enhance data by extending the extraction structure for a DataSource by adding fields in append structures.

The *Field Only Known in Exit* indicator is set for the fields of an append structure; by default these fields are not passed to the extractor from the field list and selection table.

Deselect the *Field Only Known in Exit* indicator to enable the BI Service API to pass on the append structure field to the extractor together with the fields of the delivered extract structures in the field list and in the selection table.

Enhancing the extraction structure

If you want to transfer additional information for an existing DataSource from a source system into BI, you first need to enhance the DataSource extraction structure by adding fields. To do this, create an append structure for the extraction structure (see Adding Append Structures [External]).

1. Choose *Enhance Extr. Str.,* to access field maintenance for the append structure. The name of the append structure is taken from the extraction structure name in the customer namespace.
2. Enter the fields you want to add in the field list, together with their subordinate data elements. You can use all the functions that are available for maintaining fields of tables and structures.
3. Save and activate your append.
4. Go back to the DataSource display and make sure that the *Hide Field* indicator is not selected for the newly added fields.

Function enhancement

To fill the append structure fields with data, you need to create a customer-specific function module. For information about enhancing the SAP standard with customer-specific function modules, see *Enhancing the SAP Standard* in SAP Library.

The SAP enhancement **RSAP0001** is available for enhancing BI DataSources. This enhancement contains the following enhancement components:

Transaction data	exit_saplrsap_001
Attributes, texts	exit_saplrsap_002
Hierarchies	exit_saplrsap_004

For more information, see Enhancing DataSources [Page 552].

As of Release 6.0, the Business Add-In (BAdI) **RSU5_SAPI_BADI** is available. You can display the BAdI documentation in the BAdI definition or BAdI implementation.

Application Component Hierarchy

- To create a same-level or lower-level node for a particular node, place the cursor over this node and choose *Object → Create Node.* You can also create lower-level nodes by choosing *Object → Create Children.*
- To rename, expand, or compress a node, place your cursor over the node and click on the appropriate button.
- To move a node or subtree, select the node you want to move (by positioning the cursor over it and choosing *Select Subtree)*, position the cursor on the node onto which the selected node is to be positioned. Choose *Reassign.*
- If you select a node with the cursor and choose *Set Segment,* this node is displayed with its subnodes. You can go to the higher-level nodes for this subtree using the appropriate links in the row above the subtree.

- If you select a node with the cursor and choose *Position,* the node is displayed in the first row of the view.
- All DataSources for which a valid (assigned) application component could not be found are placed under the node NODESNOTCONNECTED. The node and its subnodes are only built at transaction runtime and are refreshed when the display is saved.

  NODESNOTCONNECTED is not persistently saved to the database and is therefore not transferred in a particular state to other systems when you transport the application component hierarchy.

  Note: Hierarchy nodes created under NODESNOTCONNECTED are lost when you save. After you save, the system only displays those nodes under NODESNOTCONNECTED that were moved to this node with DataSources.

  A DataSource is positioned under an application component X. You transfer a new application component hierarchy from BI Content that does not contain application component X. In this application component, the DataSource is automatically placed under the node NODESNOTCONNECTED.

  Note: Changes to the application component hierarchy only apply until BI Content is installed again.

4.1.13.2.5.7.1 Enhancing DataSources

Use

The SAP enhancement **RSAP0001** is available if you want to fill fields that you have added to the extraction structure of a DataSource as an append structure. This enhancement is made up of the following enhancement components:

Data Type	Enhancement Component
Transaction data	exit_saplrsap_001
Attributes, texts	exit_saplrsap_002
Hierarchies	exit_saplrsap_004

See also:

Changing the SAP Standard [External] → Customer Exits [External]

Prerequisites

You have enhanced the extraction structure of the DataSource with additional fields.

Procedure

Note:

As soon as an SAP enhancement is assigned to one project, it can no longer be copied to another project.

5. In Customizing, choose the extractors (transaction SBIW in source system) *Postprocessing DataSources* → *Edit DataSources and Application Component Hierarchy*.

6. Highlight the DataSource that you want to enhance and choose *DataSource Function Enhancement*.

   The project management for SAP enhancements screen appears.

7. Specify a name for your enhancement project in the *Project* field.

8. Choose *Project → Create.*

   The *Attribute Enhancement Project <Project Name>* screen appears.

   Note:

   If a project has already been created for the SAP enhancement, use the existing project and continue with step i).

9. Enter a short description for your project.

10. Save the attributes for the project.

11. Choose *Goto → Enhancement Assignment*.

12. In the *Enhancement* field, enter the name of the SAP enhancement that you want to edit, in this case **RSAP0001**.

    You can combine several SAP enhancements in one enhancement project.

    To display the SAP documentation for an SAP enhancement, highlight the SAP enhancement and choose Goto -> Display Documentation.

13. Save your entries.

    If a project already exists for the SAP enhancement, you cannot save your entries. Go back to the initial screen and enter the existing project. Continue with step i).

14. Return to the start screen.

15. Select the C*omponent* subobject.

16. Choose *Change*.

    The system displays the SAP enhancements you have entered with the corresponding components (in this case, function exit).

    To display the documentation for a component, select the component and choose *Goto → Display Documentation*.

17. Select the component (for example, EXIT_SAPLRSAP_001) that you want to edit and choose *Edit → Select*.

    The system displays the function module prepared by the SAP application developer. Use the include program contained in this module to transfer your functionality to the module.

18. Call the include program by double-clicking on it.

    - The ABAP editor appears.

    a. Enter the source text for your function in the editor and save your include program.

    - The system asks whether you want to create an include program.

    a. Confirm that you want to create an include program.

    b. Specify the program attributes and save them.

    c. Choose *Goto → Source Code.*

       The ABAP editor appears.

d. Enter the source text for your function in the editor and save your include program.

19. Return to the start screen.

20. Activate your enhancement project by choosing *Project* → *Activate Project*.

See also:

Creating Additional Projects [External]

Creating Customer-Specific Function Modules [External]

Result

The enhancement is activated and at the runtime of the extractor, the fields that have been added to the DataSource using the append structure are filled with data.

4.1.13.3 Partner Content

Purpose

Content that consultancy partners of SAP NetWeaver Business Intelligence (BI) deliver to their customers is referred to as partner content.

The concept and technical implementation of customer and partner content largely correspond to one another. For more information about partner content, see the customer content [Page 444] documentation.

This and the following sections document only those partner content features that differ from customer content features.

Features

Partner content covers the following areas:

- Structure of the system landscape (see System Landscape [Page 445])
- Development of customer-specific content by partners
- Delivery of SAP BI Content and partner content by partners to their customers
- Option of certification for partner content through SAP

Certification of Partner Content

Certification of partner content allows you to officially document the technical quality of your content for your customers. The certification awarded by SAP is a guarantee that the partner content can be imported and installed without errors. It also ensures that SAP content objects are not inadvertently shipped again by partners.

If you want SAP to certify your content, send an e-mail to the *Integration and Certification Center* (icc@sap.com). You can purchase the contract for add-on certification from this center. The certification confirms that the following conditions are met:

- Error-free import and installation of the partner content
- Shipment of content is made using the Add-On Assembly Kit and does not contain SAP objects
- Content tested successfully by the partner

- Documentation and maintenance provided by the partner
- Release and maintenance strategies published by the partner

SAP provides generic text cases to support this testing.

When the certification process is complete, your content is awarded the *SAP certified integration* logo. Your partner content is listed in the SAP *Online Partner and Product Directory*. You can add a link with detailed information about your content.

Constraints

BI Content

You can use the BI Content delivered by SAP but you **cannot** change it.

Namespace for Developing BI Objects

- You **must** use your own namespace (see Using Namespaces for Developing BI Objects [Page 450]).
- Note also Features when Transporting Roles [Page 474]. We recommend that you use the following schema when allocating the namespace: delivery namespace /ABC/ → customer namespace /ABC/C_. This corresponds to the SAP namespaces for roles (delivery namespace SAP_BW_ → customer namespace SAP_BWC_). However, this means that you cannot create roles that begin with the letter “C”.

Delivery of BI Objects

We recommend that you use **Add-On Assembly Kits** to deliver your partner content. Using this tool, you can combine the BI objects that you have developed into one add-on. Furthermore, you can specify installation requirements (such as particular SAP releases) in the systems of your customers. You can also simplify the maintenance of your delivered content by providing your own release and support package status for each delivery. This allows you to see which content version is installed in your customer systems.

The *Add-On Assembly Kit* is not part of SAP NetWeaver and must be purchased separately. No additional tools are required in your customers’ systems: The transactions for implementing your add-ons are available in all SAP systems.

- If you want SAP to certify your content, send an e-mail to the *Integration and Certification Center* (icc@sap.com). You can use this address to purchase the *Add-On Assembly Kit* together with an add-on certification contract.
- If you do **not** want your content to be certified, you can purchase the *Add-On Assembly Kit* from the *Add-On Assembly Kit Implementation Service* (AAKIS). For more information, see *SAP Service Marketplace* at `service.sap.com/slo` → *Repository Harmonization* → *Detailed Information* → *Add-On Assembly Kit Implementation Service*.

See also:

InfoAreas, Application Components, and InfoPackage Groups [Page 505]

4.1.13.3.1 InfoAreas, Application Components, and InfoPackage Groups

Development and Delivery of Partner Content

SAP NetWeaver Business Intelligence (BI) allows you to organize objects into a tree structure by using hierarchy objects:

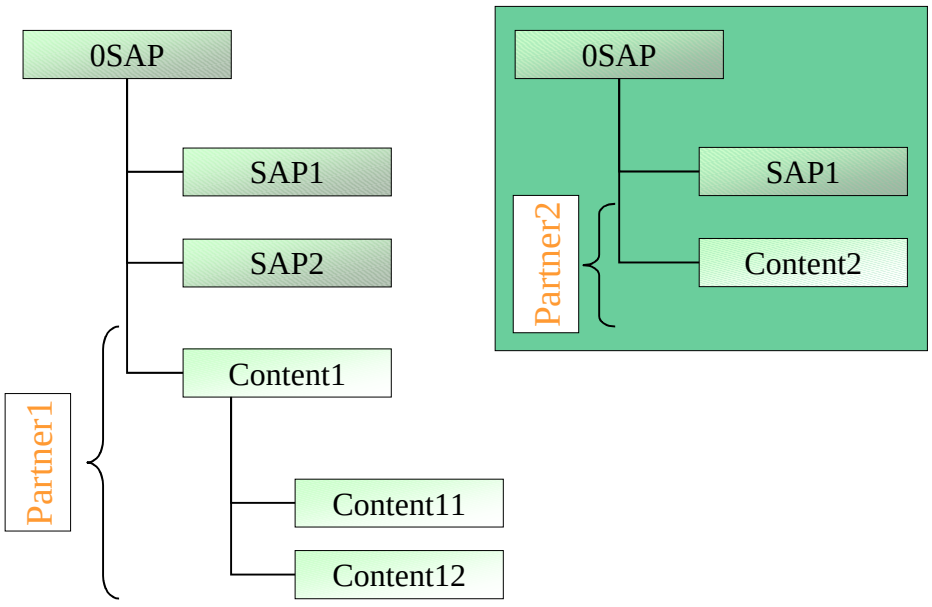
- InfoAreas [External] (in InfoObject or InfoProvider trees): for InfoObjects (in InfoObject catalogs) and InfoProviders
- Application components (in InfoSource trees): for InfoSources
- InfoPackage Groups [External] (in InfoPackage group trees): for InfoPackages. In *Show InfoPackage Groups,* you can show the InfoPackage group tree alongside the InfoSource tree.

If you want to deliver your objects along with these hierarchy objects, note the following:

In the content development system, in the *Modeling* functional area of the Data Warehousing Workbench, create exactly **one** node as a subnode for your partner content objects by using the context menu for the root node of the relevant tree. (In general this node is inserted as the last node.) You can create many additional nodes under this node.

By doing this, you ensure that the node for your partner content objects are correctly transported into the customer system.

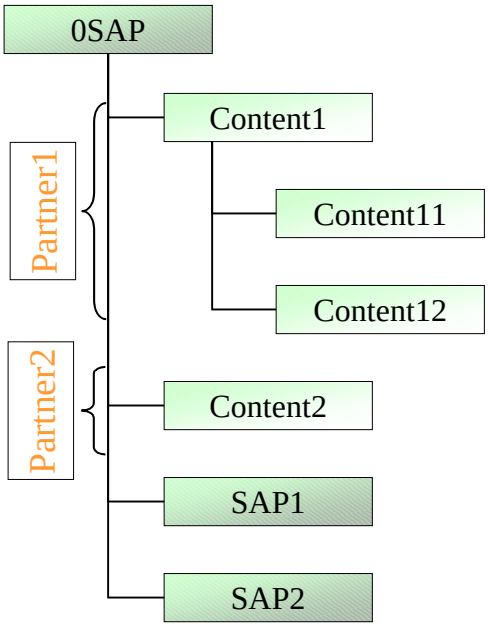
The following graphic shows two trees, in the content systems of different partners that have both been created as subnodes for the root node of a partner content node:



Import into the Customer System

The customer can import content from one or more partners. The content node is inserted under the root node for the relevant tree. If the customer imports content from various partners, these are inserted randomly. The customer can then arrange individual nodes as required.

The following graphic shows the tree in the customer system after importing content from both partners:



4.1.13.3.2 Documents for Content Objects

Use

You can create information for the content objects using the functions for document management in BI and can then deliver it with the objects.

The versioning concept ensures that no information is lost in the customer system when the partner makes repeated deliveries: If the partner delivers a new version of a document that a customer has changed on their end, the customer document is not overwritten. Instead, a new version (with its own PHIO) is delivered. The last changed version is always displayed in the customer system.

You can manually translate the documents.

Features

You can create only one document for an object. Indicate this on the *Logical Document Properties* tab page as online documentation (see Creating Documents [External]). You can also include the documents as inplace documents in the generated documentation.

4.2 Developing Persistency

Purpose

You use the following types of BI object to store data persistently in the BI system: InfoObjects, DataStore objects, and InfoCubes.

For information about using these objects, see Developing Business Logic [Page 10].

4.2.1 Differences Between Data Stores

The BI objects DataStore object and InfoCube are available for storing data. You can use these objects in different data warehouse layers. To help you in deciding when to use which object, the properties of the two types of object are compared below.

Properties of DataStore Objects and InfoCubes

	DataStore Object	InfoCube
Use	Consolidation of data in the data warehouse layer Loading delta records that can subsequently be updated to InfoCubes or master data tables Operative analysis (when being used in the operational data store)	Aggregation and performance optimization for multidimensional reporting Analytical and strategic data analysis
Type of data	Non volatile data (when being used in the data warehouse layer) Volatile data (when being used in the operational data store) Transactional data, document-type data (line items)	Non volatile data Aggregated data, totals
Type of data update	Overwrite (in rare cases: addition)	Addition only
Data structure	Flat and relational database tables, semantic key fields	Enhanced star schema (fact table and dimension tables)
Type of data analysis	Reporting at high level of granularity, flat reporting The number of query records should be strictly limited by the choice of key fields Individual document display	Multidimensional data analysis with low level of granularity (OLAP analysis) Use of InfoCube aggregates Drill-through to document level (stored in DataStore objects) possible using the report-report interface

4.3 Using Connectivity and Interoperability

Purpose

Data Extraction from SAP Source Systems

You can use BI Content extractors to transfer data into the BI system. To do this, you must first activate the BI Content DataSources shipped by SAP.

For more information, see Installing BI Content DataSources [Page 540].

You can also create DataSources for generic data extraction.

For more information, see Maintaining Generic DataSources [Page 541].

SOAP-Based Transfer of Data

You can send external data to the BI system in a controlled manner. This is known as a data push into the BI system.

For more information, see SOAP-Based Transfer of Data (3.x) [Page 561].

Transferring Data with UD Connect

UD Connect (Universal Data Connect) uses the Application Server J2EE connectivity to allow access to all relational and multidimensional data sources.

For more information, see Transferring Data with UD Connect [Page 576].

Transferring Data with DB Connect

You can use DB Connect to open up other database connections in addition to the default DB connection and to use these connections to transfer data from tables or views into a BI system.

For more information, see Data Transfer with DB Connect [Page 603].

Transferring Data from External Systems

The BI system provides open interfaces, or staging BAPIs, to enable the extraction of data and metadata from non-SAP sources at application level. These interfaces allow you to connect heterogeneous third-party tools to the BI system. You can, for example, transfer data from an Oracle application into the BI system and use it for analysis purposes in the BI system.

For more information, see Data Transfer from External Systems [Page 628].

Direct Access to Data

You can use VirtualProviders to directly access data in any source system. You can choose between various access types.

For more information, see VirtualProviders [Page 165].

Real-Time Data Acquisition

You use real-time data acquisition if you want to transfer data to BI at intervals that are more regular than scheduled data transfers (every hour or minute), and if you need to access this data frequently or regularly (several times a day, at least) for analysis purposes.

For more information, see Creating InfoPackages for Real-Time Data Acquisition [Page 647] and Creating Data Transfer Processes for Real-Time Data Acquisition [Page 649].

Open Hub Service

The open hub service allows you to distribute data from a BI system to non-SAP data marts, analytical applications, and other applications. It allows you to ensure controlled distribution across several systems. The open hub destination defines the target to which the data is relayed. For more information, see Open Hub Destination [Page 651].

Various APIs allow you to connect a third-party tool to the BI system and to use this third-party tool to distribute data to other non-SAP systems.

For more information see Third-Party Tools As Destinations [Page 654].

ICF Services

The BI system provides HTTP services to:

- Display metadata for BI objects
- Exchange metadata between different systems (see Exchanging Metadata in XMI Format [External])
- Display user-defined online documentation for metadata objects

For more information, see ICF Services in BI [Page 656].

4.3.1 Data Acquisition

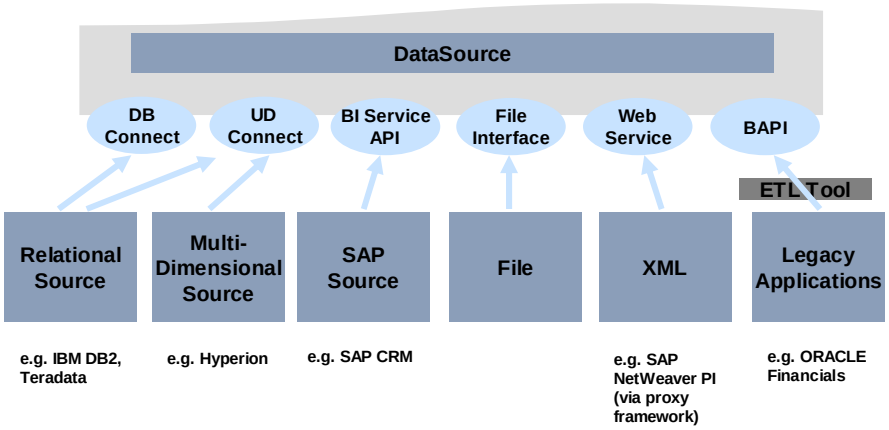
Purpose

Data retrieval is one of the data warehousing processes in BI. BI provides mechanisms for retrieving data (master data, transaction data, metadata) from various sources.

The following sections describe the sources available for the data transfer to BI and how the sources are connected to the BI system as source systems. They also describe how the data can be transferred from the sources.

The extraction and transfer of data generally occurs upon request of BI (pull). The sections about the scheduler, process chain and monitor describe how such a data request is defined and how the load process can be monitored in the BI system.

The graphic shows the sources and transfer mechanisms described below:



For more information, see:

Source System [Page 510]

Data Extraction from SAP Source Systems [Page 523]

SOAP-Based Transfer of Data [Page 561]

Transfer of Data with UD Connect [Page 576]

Transfer of Data with DB Connect [Page 603]

Transferring Data from Flat Files [Page 627]

Transferring Data from External Systems [Page 628]

4.3.1.1 Source System

Definition

All systems that provide BI with data are described as source systems. These can be:

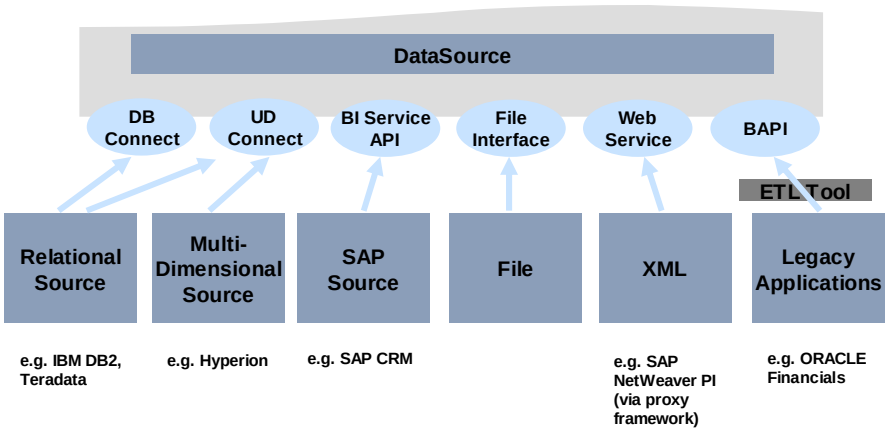
1. SAP systems
2. BI systems
3. Flat files for which metadata is maintained manually and transferred to BW using a file interface
4. Database management systems into which data is loaded from a database supported by SAP using DB Connect, without using an external extraction program
5. Relational or multidimensional sources that are connected to BI using UD Connect
6. Web Services that transfer data to BI by means of a push
7. Non-SAP systems for which data and metadata is transferred using staging BAPIs.

You define source systems in the Data Warehousing Workbench in the source system tree. To define a source system, choose *Create* in the context menu of the folder in the source system type.

Integration

DataSources are used to extract and stage data from source systems. The DataSources subdivide the data that is provided by a source system into self-contained business areas.

The following graphic provides an overview of the data transfer sources supported by BI and shows the interfaces that you can use:



4.3.1.1.1 Connection between Source Systems and BI

As a data warehouse, BI is extensively networked with other systems. There are usually several source systems connected to a single BI, and a BI system itself can also be used as a source system. In this case, we speak of data marts [External].

Since changes to a system in the BI source system network affect all of the systems connected to the BI, systems cannot be treated or viewed in isolation.

A connection between a source system and a BI system consists of a sequence of individual links and settings that are made in both systems.

- RFC connections
- ALE settings
  - Partner profiles
  - Port
  - IDoc types
  - IDoc segments

- BI settings

Refer to the BI Customizing Implementation Guide under *Business Information Warehouse → Connections to Other Systems* for details on the customizing settings that are relevant when connecting source systems to a BI system.

See also:

Logical System Names [External]

4.3.1.1.2 Creating SAP Source Systems

Prerequisites

You have defined the necessary configuration in BI and in the source system.

See also:

Configurations in the BI [Page 513]

Configurations in the SAP Source Systems [Page 514]

Procedure

8. In the source system tree in the Data Warehousing Workbench, choose *Create* in the context menu of the *BI* or *SAP* folder.
9. If the destination already exists for the SAP source system, select the destination.
10. If no destination exists, enter the server name information from the source system.

    Application server: pswdf090

    System ID: IDF

    System number: 90
11. Enter a password for the background user in the source system and confirm this in the next input row. If the user already exists in the source system, enter the valid password.
12. Enter the password that you defined for the BI background user in the Implementation Guide under *SAP NetWeaver → Business Intelligence → Links to other Systems → Link between SAP Systems and BW → Maintain proposal for users in the source system (ALE communication)*, and confirm.
13. Choose *Transfer.* You get to the remote login in the source system.
14. Select the relevant clients and logon as a system administrator. First make sure that you have the authority to create users and RFC destinations.

    The RFC destinations for BI and the background users are thus created automatically in the source system. If the RFC destination already exists in the source system, check its accuracy. You can test the RFC destination using the function *Test → Links* and *Test → Authorizations*. For more information on RFC destinations, see Maintaining Remote Destinations [External].

    User profiles are also mapped automatically. If the user already exists in the source system, check the accuracy of the profile [External]

If it does not already exist, the RFC destination for the source system is now created automatically in BI, with the information read in the source system .

Result

The ALE settings, which are needed for the communication between a BI System and an SAP System, are created in the background with the use of the created destinations. These settings are made in BI as well as in the source system. The BI settings for the new connection are created in BI.

If the new SAP source system has been created, metadata is requested automatically from the source system. The metadata for DataSources is also replicated to BI in the D version.

4.3.1.1.2.1 Configurations in BI

Make the settings described below for each SAP source system you want to connect to the BI system in order to transfer data.

IMG Settings for Connecting to Other SAP Systems

In the Implementation Guide (IMG) under *SAP NetWeaver → Business Intelligence → Connections to Other Systems* make the following settings:

- General connection settings
  - Define logical system
  - Assign logical system to client
- Connections between SAP systems and the BI system
  - Maintain the suggestion for user in source system (ALE communication)
- Perform automatic workflow customizing

The name you used for the source system (logical system) has to be used again when creating the source system in the Data Warehousing Workbench.

Settings for the System Change Option

As a rule, system changes are not permitted in productive systems. Connecting a system as a source system to a BI system or connecting a BI system to a new source system will, however, mean changes as far as the system change option is concerned. Make the following settings in the BI system to ensure that the following changes are valid in the relevant clients of the systems when connecting the source system.

- **Cross-client Customizing and repository changes**

  In the Implementation Guide under *SAP NetWeaver → Business Intelligence → Links to Other Systems → General Connection Settings → Assign Logical System to Client*, select the relevant clients under *Goto → Details*.

  Choose the entry *Changes to Repository and Cross-Client Customizing Permitted* in the field *Changes to Cross-Client Objects.*

- **Changes to the software components *Local Developments* (LOCAL) and *SAP NetWeaver BI* (SAP_BW)**

  In the Transport Organizer Tools (Transaction SE03) choose *Administration → Set System Change Option* and then *Execute*. In the next screen mark the software components LOCAL and SAP_BW as *changeable*.

- **Changes to the customer name range.**

  In the Transport Organizer Tools (Transaction SE03) choose *Administration → Set System Change Option* and then *Execute*. In the next screen mark the customer namespace as *changeable*.

- **Changes in the BI namespaces with prefix `/BI0/` (SAP namespace) and `/BIC/` (customer namespace)**

  In the Transport Organizer Tools (Transaction SE03) choose *Administration → Set System Change Option* and then *Execute*. On the next screen, mark the BI namespaces with the prefixes /BI0/ and /BIC/ as *changeable*.

4.3.1.1.2.2 Configurations in the SAP Source System

Make the settings described below once for each SAP source system you want to connect to a BI system in order to transfer data.

IMG Settings for Connecting to a BI System

In the Implementation Guide (IMG) under SAP NetWeaver *→ SAP Web Application Server → IDoc-Interface / Application Link Enabling (ALE) → Basic Settings* make the following settings:

- Logical Systems
  - Define logical system
  - Assign logical system to client
- Perform automatic workflow customizing

The name you used for the source system (logical system) has to be used again when creating the source system in the Data Warehousing Workbench of the BI system.

Settings for the System Change Option

As a rule, system changes are not permitted in productive systems. Connecting a system as a source system to a BI system or connecting a BI system to a new source system will, however, mean changes as far as the system change option is concerned. Make the following settings in the source system to ensure that the following changes are valid in the relevant clients of the systems when connecting the source system.

- **Cross-client Customizing and repository changes**

  In the Implementation Guide under *SAP NetWeaver → SAP Business Information Warehouse → Links to Other Systems → General Connection Settings → Assign Logical System to Client*, select the relevant clients under *Goto → Details*.

  In the *Cross-Client Object Changes* field, choose the *Changes to Repository and Cross-Client Customizing Allowed* option.

- **Changes to the software components *Local Developments* (`LOCAL`) and *SAP NetWeaver BI* (`SAP_BW`)**

  In the Transport Organizer Tools (Transaction SE03) choose *Administration → Set System Change Option* and then *Execute*. In the next screen mark the software components `LOCAL` and `SAP_BW` as *changeable*.

- **Changes to the customer name range.**

In the Transport Organizer Tools (Transaction SE03) choose *Administration → Set System Change Option* and then *Execute*. In the next screen mark the customer namespace as *changeable*.

- **Changes in the BI namespaces with prefix `/BI0/` (SAP namespace) and `/BIC/` (customer namespace)**

  In the Transport Organizer Tools (Transaction SE03) choose *Administration → Set System Change Option* and then *Execute*. On the next screen, mark the BI namespaces with the prefixes /BI0/ and /BIC/ as *changeable*.

Determining the Server Name

When you connect a SAP source system to a BI system, you define the server that should be used for the source system connection. To get the server name, choose *Tools → Administration → Monitor → System Monitoring → Server.* The server name is displayed as follows: server_<SAPSID>_<instance_no>, e.g. pswdf090_IDF_90

4.3.1.1.2.3 Transferring Global Settings

Use

With this function you are able to transfer various types of table content from connected SAP source systems. The types of table content are units of currency, units of measure, fiscal year variants and factory calendars.

Prerequisites

The relevant tables have already been maintained in the SAP source system.

Procedure

1. Under *Modeling*, choose the source system tree in the Administrator Workbench.
2. Select your SAP source system and choose *Transfer Global Settings* from the context menu. You reach the *Transfer Global Settings: Selection* screen.
3. Under *Transfer Global Table Contents,* select the settings that you want to transfer:
   - Currencies to transfer all settings relevant to currency translation from the source system. For more information, see Transferring Global Table Contents for Currencies from SAP Systems [External].
   - Units of measure to transfer settings for units of measure from the source system.
   - Fiscal year variants to transfer settings for fiscal year variants from the source system.
   - Factory calendars to transfer settings for factory calendars from the source system.
4. Under *Mode* you can specify whether the upload should just be simulated, and whether the settings are to be updated or transferred again. With the *Update Tables* option, existing records are updated. With the *Rebuild Tables* option, the corresponding tables are deleted before the new records are loaded.

5. Choose *Execute*.

4.3.1.1.3 Creating External Systems

Prerequisites

1. You have made the following settings in the BW Customizing Implementation Guide under *Business Information Warehouse* → *Connections to other Systems*.
   - General connection settings
   - Verify workflow Customizing
2. As a rule, system changes are not permitted in productive systems. Connecting a system as a source system to BI, or connecting BI to a new source system will, however, mean changes as far as the system change option is concerned. For the clients concerned in the BI system therefore, you have made sure that the following changes are permitted during the source system connection.
   - Cross-client Customizing and repository changes

     In the BW Customizing Implementation Guide, select the relevant client under *Business Information Warehouse* → *Connections to Other Systems* → *General Connection Settings* → *Assign Logical System to Client*, then choose *Goto* → *Detail*. In the *Cross-Client Object Changes* field, choose the *Changes to Repository and Cross-Client Customizing Allowed* option.
   - Changes to the local developments and Business Information Warehouse software components

     You use transaction SE03 (Organizer Tools) to set the change options. Choose *Organizer Tools* → *Administration* → *Set Up System Change Option*, then *Execute*. Make the following settings on the next screen.
   - Changes to the customer name range.

     Again, you use transaction SE03 to set the change option for the customer name range.
   - Changes to BI namespaces /BIC/ and /BI0/

     Again, you can set up the changeability of the BI namespace with transaction SE03.

Procedure

1. In the source system tree in the Data Warehousing Workbench, choose *Create* in the context menu of the *External System* folder.
2. Enter a name and a description.
3. For your extraction tool, maintain the destination that is to be referred to when loading data from BI.

Result

When you use the created destinations, the ALE settings that are necessary for communication between a BI system and an external system are created in BI in the background. The BI settings for the new connection are created in BI.

See also:

Maintaining InfoSources (External System) [Page 300]

4.3.1.1.4 Creating File Systems

Prerequisites

1. You have made the following settings in the BW Customizing Implementation Guide under *Business Information Warehouse → Connections to other Systems*.
   - General connection settings
   - Connection between flat files and BI
   - Verify workflow Customizing
2. As a rule, system changes are not permitted in productive systems. Connecting a system as a source system to BI, or connecting BI to a new source system will, however, mean changes as far as the system change option is concerned. For the clients concerned in the BI system therefore, you have made sure that the following changes are permitted during the source system connection.
   - Cross-client Customizing and repository changes

     In the BW Customizing Implementation Guide, select the relevant client under *Business Information Warehouse → Connections to Other Systems → General Connection Settings → Assign Logical System to Client*, then choose *Goto → Detail*. In the *Cross-Client Object Changes* field, choose the *Changes to Repository and Cross-Client Customizing Allowed* option.
   - Changes to the local developments and Business Information Warehouse software components

     You use transaction SE03 (Organizer Tools) to set the change options. Choose *Organizer Tools → Administration → Set Up System Change Option*, then *Execute*. Make the following settings on the next screen.
   - Changes to the customer name range.

     Again, you use transaction SE03 to set the change option for the customer name range.
   - Changes to BI namespaces /BIC/ and /BI0/

     Again, you can set up the changeability of the BI namespace with transaction SE03.

Procedure

1. In the source system tree in the Data Warehousing Workbench, choose *Create* in the context menu of the *File* folder.

2. Enter the technical name of your file system under *Source system* and enter a description.

Result

In BI, the ALE settings that are necessary for communication between a BI system and a file system are created in the background. The BI settings for the new connection are created in BI.

See also:

Maintaining InfoSources (Flat Files) [Page 286]

4.3.1.1.5 Creating Database Management Systems as Source Systems

Use

With DB Connect you have the option of opening extra database connections in addition to the SAP default connection. You use these connections during extraction to BI to access databases and transfer data into a BI system. To do this, you have to create a database source system in which the connection data is specified and made known to the ABAP runtime environment. The connection data is used to identify the source database and authenticate the database.

Prerequisites

- You have made the following settings in the Implementation Guide (IMG) under *SAP NetWeaver → Business Intelligence → Connections to Source Systems*:
  - General connection settings
  - Perform automatic workflow customizing
- As a rule, system changes are not permitted in productive systems. Connecting a system to BI as a source system, or connecting BI to a new source system, represents a change to the system. Therefore, you have to ensure that in the clients of the BI system that are affected, the following changes are permitted during the source system connection.
  - Cross-client Customizing and repository changes

    In the Implementation Guide (IMG) under *SAP NetWeaver → Business Intelligence → Links to Source Systems → General Connection Settings → Assign Logical System to Client*, select the relevant clients and choose *Goto → Details*. In the *Cross-Client Object Changes* field, choose the *Changes to Repository and Cross-Client Customizing Allowed* option.
  - Changes to the local developments and Business Information Warehouse software components

    You use transaction SE03 (Organizer Tools) to set the change options. Choose *Organizer Tools → Administration → Set Up System Change Option*. Choose *Execute*. On the next screen, make the following settings:.
  - Changes to the customer name range.

Again, you use transaction SE03 to set the change option for the customer name range.

- Changes to BI namespaces /BIC/ and /BI0/

  Again, use transaction SE03 to set the changeability of the BI namespace.

- If the source DBMS and BI DBMS are different:
  - You have installed the database-specific DB client software on your BI application server. You can get information about the database-specific DB client from the respective database manufacturers.
  - You have installed the database-specific DBSL on your BI application server [Page 606].
- In the database system, you have created a username and password that you want to use for the connection.

  See Database Users and Database Schemas [Page 610].

Procedure

Before you can open a database connection, all the connection data that is used to identify the source database and authenticate the database has to be made known to the ABAP runtime environment. For this, you need to specify the connection data for each of the database connections that you want to set up in addition to the SAP default connection.

1. In the source system tree in the Data Warehousing Workbench, choose *Create* in the context menu of the *DB Connect* folder.
2. On the following screen, specify the logical system name (= DB connection) and a descriptive text for the source system. Choose *Continue*.

   The *Change "Description of Database Connection" View: Detail* screen appears.
3. Select the database management system (DBMS) that you want to use to manage the database. This entry determines the database platform for the connection.
4. Under *User Name,* specify the database user under whose name you want the connection to be opened.
5. When establishing the connection, enter the user *DB Password* twice for authentication by the database. This password is encrypted and stored.
6. Under *Connection Info*, specify the technical information required to open the database connection. This information, which is needed when you establish a connection using NATIVE SQL, depends on the database platform and encompasses the database names and the database host on which the database runs. he string informs the client library of the database to which you want to establish the connection.

Connection information that depends on the database platform

Supported Database	CON_ENV Connection Information
SAP DB (ada) or MaxDB (dbs)	<server_name>-<db_name>

Microsoft SQL Server (mss)	MSSQL_SERVER=<server_name> MSSQL_DBNAME=<db_name> MSSQL_SERVER=10.17.34.80 MSSQL_DBNAME=Northwind (See SAP Note 178949 - MSSQL: Database MultiConnect with EXEC SQL)
Oracle (ora)	TNS Alias (See SAP Note 339092 - DB-MultiConnect with Oracle as a secondary database)
DB2/390 (db2)	 PORT=4730;SAPSYSTEMNAME=D6B;SSID=D6B0;SAPSYSTEM=71;SAPDBHOST=ihsapfc; ICLILIBRARY=/usr/sap/D6D/SYS/exe/run/ibmiclic.o The parameters describe the target system for the connection (see installation handbook DB2/390). The individual parameters (PORT=... SAPSYSTEMNAME=... .....) must be separated with **' '** , **','** or **';'**. (See SAP Note 160484 - DB2/390: Database MultiConnect with EXEC SQL)
DB2/400 (db4)	<parameter_1>=<value_1>;...;<parameter_n>=<value_n>; You can specify the following parameters: • AS4_HOST: Host name for the Remote DB Server. You have to enter the host name in the same format as is used under TCP/IP or OptiConnect, according the connection type you are using. You have to specify the AS4_HOST parameter. • AS4_DB_LIBRARY: Library that the DB server job needs to use as the current library on the remote DB server. You have to enter parameter AS4_DB_LIBRARY. • AS4_CON_TYPE: Connection type; permitted values are OPTICONNECT and SOCKETS. SOCKETS means that a connection is used using TCP/IP sockets. Parameter AS4_CON_TYPE is optional. If you do not enter a value for this parameter, the system uses connection type SOCKETS. For a connection to the remote DB server as0001 on the RMTLIB library using TCP/IP sockets, you have to enter: **AS4_HOST=as0001;AS4_DB_LIBRARY=RMTLIB;AS4_CON_TYPE=SOCKETS;** The syntax must be exactly as described above. You cannot have any additional blank spaces between the entries and each entry has to end with a semicolon. Only the optional parameter AS4_CON_TYPE=SOCKETS can be omitted. (See SAP Note 146624 - AS/400: Database MultiConnect with EXEC SQL) (For DB MultiConnect from Windows AS to iSeries, see Note 445872)

DB2 UDB (db6)	**DB6_DB_NAME=<db_name>** , where **<db_name>** is the name of the DB2 UDB database on which you want to run Connect. You want to establish a connection to the ‘mydb’ database. Enter **DB6_DB_NAME=mydb** as the connection information. (See SAP Note 200164 - DB6: Database MultiConnect with EXEC SQL)

7. Specify whether your database connection needs to be permanent or not.

   If you set this indicator, losing an open database connection (for example due to a breakdown in the database itself or in the database connection [network]) has a negative impact.

   Regardless of whether this indicator is set, the SAP work process tries to reinstate the lost connection. If this fails, the system responds as follows:

   a. The database connection is **not permanent**, which means that the indicator is **not** set:

      The system ignores the connection failure and starts the requested transaction. However, if this transaction accesses the connection that is no longer available, the transaction terminates.

   b. The database connection is **permanent**, which means that the indicator is set:

      After the connection terminates for the first time, each transaction is checked to see if the connection can be reinstated. If this is not possible, the transaction is not started – independently of whether the current transaction would access this special connection or not. The SAP system can only be used again once all the permanent DB connections have been reestablished.

      We recommend setting the indicator if an open DB connection is essential or if it is accessed often.

8. Save your entry and go back.
9. The *Change “Description of Database Connections” View: Overview* screen appears. The system displays the entry for your database connection in the table.
10. Go back.

Result

You have created IDoc basic types, port descriptions, and partner agreements. When you use the destinations that you have created, the ALE settings that enable a BI system to communicate with a database source system are created in BI in the background. In addition, the BI settings for the new connection are created in the BI system and the access paths from the BI system to the database are stored.

You have now successfully created a connection to a database source system. The system displays the corresponding entry in the source system tree. You can now create DataSources for this source system.

4.3.1.1.6 Creating a UD Connect Source System

Prerequisites

On the J2EE Engine of a SAP system, you have set the connection to the data source with your relational or multidimensional source objects.

You have created the RFC destinations on the J2EE Engine (of a SAP system) and in BI in order to enable communication between the J2EE Engine and BI. For more information, see the Implementation Guide for *SAP NetWeaver → Business Intelligence → UDI Settings by User Scenarios → UD Connect Settings*.

Procedure

1. In the source system tree in the Data Warehousing Workbench, choose *Create* in the context menu of the *UD Connect* folder.
2. Select the required *RFC Destination* for the J2EE Engine.
3. Specify a logical system name.
4. Select the type of connector.
5. Select the name of the connector.
6. Specify the name of the source system if it was not determined from the logical system name.
7. Choose *Continue*.

Result

When you use the created destinations, the settings that are required for communication between BI and the J2EE are created in BI.

4.3.1.1.7 Check Source System

Use

The source system check for correct configuration includes the

- The RFC connection
- The ALE settings, and
- The BW settings

in relation to BW and source system.

Errors in the configuration are displayed in a log.

Activities

To do this, choose *Source System Tree → Your Source System → Context menu (right mouse button) → Check* in the BW Administrator Workbench - Modeling.

See also:

Connection between Source System and BW [Page 511]

4.3.1.2 Data Extraction from SAP Source Systems

Purpose

Extractors are part of the data retrieval mechanisms in the SAP source system. An extractor can fill the extraction structure of a DataSource with the data from SAP source system datasets.

Replication makes the DataSource and its relevant properties known in BI.

For the data transfer to the input layer of BI, the Persistent Staging Area (PSA), define the load process with an InfoPackage in the scheduler. The data load process is triggered by a request IDoc to the source system when the InfoPackage is executed. We recommend that you use process chains for execution.

Process Flow

There are application-specific extractors, each of which is hard-coded for the DataSource that was delivered with BI Content, and which fill the extraction structure of this DataSource.

In addition, there are generic extractors with which you can extract more data from the SAP source system and transfer it into BI. Only when you call up the generic extractor by naming the DataSource does it know which data is to be extracted, and from which tables it should read it from and in which structure. This is how it fills different extraction structures and DataSources.

You can run generic data extraction in the SAP source system application areas such as LIS, CO-PA, FI-SL and HR. This is how LIS, for example, uses a generic extractor to read info structures. DataSources are generated on the basis of these (individually) defined info structures. We speak of customer-defined DataSources with generic data extraction from applications.

Regardless of application, you can generically extract master data attributes or texts, or transaction data from all transparent tables, database views or SAP query functional areas or using the function module. You can generate user-specific DataSources here. In this case, we speak of generic DataSources.

The DataSource data for these types are read generically and transferred into BI. This is how generic extractors allow the extraction of data that cannot be made available within the framework of BI Content.

PlugIn for SAP Systems

BI-specific source system functions, extractors and DataSources are delivered for specific SAP systems by plug-ins.

Communication between the SAP source system and BI is only possible if the appropriate plug-in is installed in the source system.

4.3.1.2.1 DataSource in the SAP Source System

Definition

Data that logically belongs together is stored in the source system in the form of DataSources.

A DataSource consists of a quantity of fields that are offered for data transfer into BI. The DataSource is technically based on the fields of the extraction structure [Page 524]. By defining a DataSource, these fields can be enhanced as well as hidden (or filtered) for the data transfer.

Additionally, the DataSource describes the properties of the associated extractor with regard to data transfer to BI. Upon replication, the BI-relevant properties of the DataSource are made known in BI.

Integration

DataSources are used for extracting data from an SAP source system and for transferring data into BI. DataSources make the source system data available to BI on request in the form of the (if necessary, filtered and enhanced) extraction structure. In the DataSource maintenance in BI, you determine which fields from the DataSource are actually transferred. Data is transferred in the input layer of BI, the Persistent Staging Area (PSA). In the transformation, you determine what the assignment of fields from the DataSource to InfoObjects from BI should look like. Data transfer processes facilitate the further distribution of the data from the PSA to other targets. The rules that you set in the transformation apply here.

4.3.1.2.1.1 Extraction Structure

Definition

In the extraction structure, data from a DataSource is staged in the source system. It contains the amount of fields that are offered by an extractor in the source system for the data loading process.

You can edit DataSource extraction structures in the source system. In particular, you can determine the DataSource fields in which you hide extraction structure fields from the transfer. This means filtering the extraction structure and/or enhancing the DataSource for fields, meaning completing the extraction structure. In transaction SBIW in the source system, choose *Business Information Warehouse → Subsequent Processing of DataSources.*

4.3.1.2.2 Transferring BI Content DataSources into Active Version

Use

The DataSources delivered by SAP with BI Content are in the source system in the SAP delivery version (D version). If you want to transfer data from a source system into BI using a BI Content DataSource, you have to first transfer the data from the D version into the active version (A version).

BI Content DataSources of a source system are only available to you in BI for transferring data, if you have transferred these in their active versions in the source system and then replicated them in BI.

Procedure

- To transfer and activate a DataSource delivered by SAP with Business Content, in transaction SBIW in the source system choose *Business Information Warehouse → Business Content DataSources or Activating SAP Business Content → Transfer Business Content DataSources*. For more information on this activity, see the documentation for customizing extractors.
- When activating BI Content in BI, you can also activate DataSources remotely from the BI system. This activation is subject to an authorization check. You need role SAP_RO_BCTRA. Authorization object S_RO_BCTRA is checked. The authorization is valid for all DataSources of a source system.

  Remote activation is not supported in source systems and/or in BI systems with a BI Service API release lower than SAP NetWeaver 2004s. In this case, you have to activate the DataSources in the source system manually and then replicate them to the BI system.

  For information on the procedure, see Installing Business Content [Page 435]. See the information in Collection Object Grouping [External] in the **Features** section → **Data Flow Before**.

4.3.1.2.3 Editing the DataSource in the Source System

Use

You can edit DataSources in the source system, using transaction SBIW.

For more information on maintaining DataSources, choose *Subsequent Processing of DataSources → Edit DataSource* in transaction SBIW.

4.3.1.2.4 Replication of DataSources

Use

In the SAP source system, the DataSource is the BI-relevant metaobject that makes source data available in a flat structure for data transfer into BI. In the source system, a DataSource can have the SAP delivery version (D version: Object type R3TR OSOD) or the active version (A version: Object type R3TR OSOA).

The metadata from the SAP source systems is not dependent on the BI metadata. There is no implicit assignment of objects with the same names. In the source system, information is only retained if it is required for data extraction. Replication allows you to make the relevant metadata known in BI so that data can be read more quickly. The assignment of source system objects to BI objects takes place exclusively and centrally in BI.

There are two types of DataSources in BI. A DataSource can either be a DataSource (R3TR RSDS) or a 3.x DataSource (R3TR ISFS). Since a DataSource cannot exist simultaneously in both object types in one source system and because these objects are not differentiated in the system, you have to choose which object you want the metadata to be replicated in when you replicate the DataSource.

Integration

Replicated 3.x DataSources can be used as emulated DataSources in BI for migration purposes. As long as certain prerequisites are fulfilled, a 3.x DataSource can be restored from a DataSource.

See also:

Emulation, Migration and Restoring DataSources [Page 37]

Prerequisites

You have connected the source system to BI correctly.

Features

Metadata from a SAP source system is read in the source system using the *Replicate Metadata* function in the DataSource tree of the Data Warehousing Workbench and is replicated to BI or created there. When you create a SAP source system, an automatic metadata upload is performed.

In the DataSource tree, you can request the metadata for a DataSource, the metadata for an application component, or all metadata for a source system: The *Replicate Tree Metadata* function replicates the application component hierarchy of the SAP source system to BI.

Replication Process Flow

In the **first step**, the **D versions** are replicated.

The DataSource header tables of BI Content DataSources are saved in BI as the D version. Replicating the header tables is a prerequisite for collecting and activating BI Content.

- If SHDS is available for the D-TLOGO object in the BI shadow content, the relevant metadata is replicated in the DataSource (R3TR RSDS).
- If SHMP (mapping for 3.x DataSource) is available for the D-TLOGO object in the BI shadow content, the relevant metadata is replicated in the DataSource (R3TR ISFS).
- If no BI Content is available in the D version in BI for a DataSource (R3TR OSOD), the D version cannot be replicated because this version is only used in BI for BI Content activation.

In the **second step**, the **A versions** are replicated.

DataSources (R3TR RSDS) are saved in the M version in BI with all relevant metadata. This means that as long as the DataSource is not yet being used, (as long as a transformation does not yet exist for the DataSource), you can avoid generating too many DDIC objects needlessly.

DataSources (R3TR ISFS) are saved in BI in the A version with all the relevant metadata.

- As a basic principle, the object type of the A version follows the object type of the D version. If the DataSource already exists in BI in the A or D version, the DataSource is replicated to the existing object.
- If the DataSource does not yet exist in BI, the system performs replication according to the following logic:
    a. If the DataSource is a hierarchy or export DataSource, this determines the object type for the replication:
        - Hierarchy DataSources are replicated to 3.x DataSources.
        - Export DataSources (8*) are replicated to 3.x DataSources.

b. If there is a D version in BI for a mapping object (R3TR ISMP), the system performs replication to 3.x DataSource (R3TR ISFS).

c. Otherwise, the system asks the user to which object type the DataSource is to be replicated.

Note that you replicate the DataSource correctly: For example, if you have modeled the data flow with 3.x objects from BI Content and used update and transfer rules, make sure that you replicate the DataSource to a 3.x DataSource. If you replicated the DataSource incorrectly, you can no longer use the BI Content data model.

Deleting During Replication

DataSources are only deleted during replication if you perform replication for an entire source system or for a particular DataSource. When you replicate DataSources for a particular application component, the system does not delete any DataSources because they may have been assigned to another application component in the meantime.

If the system determines upon replication that the D version of a DataSource in the source system or the associated BI Content (shadow objects of DataSource R3TR SHDS or shadow objects of mapping R3TR SHMP) is not (or is no longer) available in BI, the system automatically deletes the D version in BI.

If the system determines upon replication that the A version of a DataSource in the source system is not (or is no longer) available, the BI system asks whether you want to delete the DataSource in BI. If you confirm that you want to delete the DataSource, the system also deletes all dependent objects, the PSA, InfoPackage, transformation, data transfer process (where applicable) and, in the case of 3.x DataSource, the mapping and transfer structure, if these exist.

Before confirming that you want to delete the DataSource and related objects, ensure that you are no longer using the objects that will be deleted. If it is only temporarily not possible to replicate the DataSource, confirming the deletion prompt may cause relevant objects to be deleted.

Replication with Data Request

When data is requested from BI, the DataSource is replicated automatically if it has been changed in the source system. A distinction must be made between DataSource types and the types of changes in the source system.

This function ensures that requests are not refused in the source system because of the default time stamp comparison even though the DataSource has not changed substantially.

DataSource (R3TR RSDS)

When a request is instantiated in the InfoPackage, the DataSource is refreshed in BI if the DataSource in the source system has a more recent time stamp than the DataSource in BI. In addition, the DataSource is activated in BI (including transfer structure generation in the source system) if it is older than the DataSource in the source system. However, it is only activated if the object status is “active“ after replication.

This is **not** the case if changes have been made in the source system to the field property (name, length, type) or if a field has been excluded from the transfer (because, for example, the *Hide Field* indicator is set in the field list of the DataSource or the field property has been changed in the extraction structure). In these cases, the DataSource is deactivated in BI.

If the DataSource is not active after replication, the system produces an error message. The DataSource must be activated manually.

3.x DataSource (R3TR ISFS)

When a request is instantiated in the InfoPackage, the DataSource replicate is refreshed in BI if the DataSource in the source system has a more recent time stamp than the DataSource replicate in BI. In addition, the transfer structure is activated in BI if it is older than the DataSource in the source system. However, it is only activated if the object status is “active“ after replication.

This is **not** the case if changes have been made in the source system to the field property (name, length, type) or if a field has been excluded from the transfer (because, for example, the *Hide Field* indicator is set in the field list of the DataSource or the field property has been changed in the extraction structure). In these cases, the transfer structure is deactivated in BI.

If the transfer structure is not active after replication because, for example, a field property has been changed, no transfer structure exists, or the transfer structure has been deactivated because of changes to the data flow, the system produces an error message; the transfer structure has to be activated manually.

4.3.1.2.5 Editing DataSources from SAP Source Systems in BI

Use

A DataSource is defined in the SAP source system along with its properties and field list. In DataSource maintenance in BI, you determine which fields of the DataSource are to be transferred to BI. In addition, you can change the properties for extracting data from the DataSource and properties for the DataSource fields.

Prerequisites

You have replicated the DataSource in BI.

Procedure

You are in an object tree in the Data Warehousing Workbench.

1. Select the required DataSource and choose *Change.*
2. Go to the *General* tab page.

   Select *PSA Not Typed* if you do not want to generate the PSA for the DataSource in a typed structure but want to use character-type fields of type CHAR exclusively.

   Use this option if conversion during loading causes problems, for example, because there is no appropriate conversion routine, or if the source cannot guarantee that data is loaded with the correct data type.

   In this case, data can be loaded into the PSA after the DataSource has been activated and be corrected there.
3. Go to the *Extraction* tab page.

a. Under *Adapter*, you determine how the data is to be accessed. The options depend on whether the DataSource supports direct access and real-time data acquisition.

b. Select *Number Format* → *Direct Entry* to specify the character for the thousand separator and the decimal point character that you want to use for the DataSource fields. If a *User Master Record* has been specified, the system applies the settings of the user who is used when the conversion exit is executed. This is usually the BI background user (see also: User Management [External]).

4. Go to the *Fields* tab page.

   a. Under *Transfer*, specify the decision-relevant DataSource fields that should be available for extraction and transferred to BI.

   b. If required, change the setting for the *Format* of the field.

   c. If you choose an *External Format*, ensure that the output length of the field (external length) is correct. Change the entries, as required.

   d. If required, specify a conversion routine that converts data from an external format into an internal format.

   e. Under *Currency/Unit*, change the entries for the referenced currency and unit fields as required.

5. Check, save and activate your DataSource.

Result

When you activate the DataSource, BI generates a PSA table and a transfer program.

You can now create an InfoPackage. You define the selections for the data request in the InfoPackage. The data can be loaded into the entry layer of the BI system, the PSA. Alternatively you can access the data directly if the DataSource supports direct access and you have a VirtualProvider in the definition of the data flow.

4.3.1.2.6 Using Emulated 3.x DataSources

Use

You can display an emulated 3.x DataSource in DataSource maintenance in BI. Changes are not possible in this display. In addition, you can use emulation to create the (new) data flow for a 3.x DataSource with transformations, without having to migrate the existing data flow that is based on the 3.x DataSource.

We recommend that you use emulation before migrating the DataSource in order to model and test the functionality of the data flow with transformations, without changing or deleting the objects of the existing data flow. Note that use of the emulated Data Source in a data flow with transformations has an effect on the evaluation of the settings in the InfoPackage. We therefore recommend that you only use the emulation in a development or test system.

Constraints

An emulated 3.x DataSource does not support real-time data acquisition, using the data transfer process to access data directly, or loading data directly (without using the PSA).

Prerequisites

If you want to use transformations in the modeling of the data flow for the 3.x DataSource, the transfer rules and therefore the transfer structure must be activated for the 3.x DataSource. The PSA table to which the data is written is created when the transfer structure is activated.

Procedure

To display the emulated 3.x DataSource in DataSource maintenance, highlight the 3.x DataSource in the DataSource tree and choose *Display* from the context menu.

To create a data flow using transformations, highlight the 3.x DataSource in the DataSource tree and choose *Create Transformation* from the context menu. You also use the transformation to set the target of the data transferred from the PSA.

To permit a data transfer to the PSA and further updating of the data from the PSA to the InfoProvider, select the DataSource 3.x in the DataSource tree and choose *Create InfoPackage* or *Create Data Transfer Process* in the context menu. We recommend that you use the processes for data transfer to prepare for the migration of a data flow and not in the production system.

Result

If you defined and tested the data flow with transformations using the emulation, you can migrate the DataSource 3.x after a successful test.

4.3.1.2.7 Data Reconciliation

Purpose

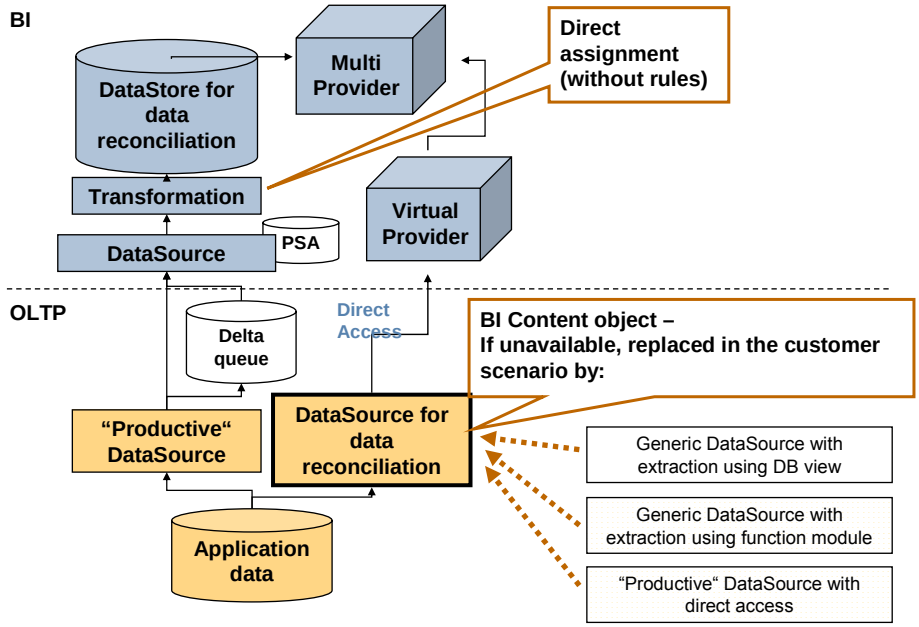
An important aspect in ensuring the quality of data in BI is the consistency of the data. As a data warehouse, BI integrates and transforms data and stores it so that it is made available for analysis and interpretation. The consistency of the data between the various process steps has to be ensured. Data reconciliation for DataSources allows you to ensure the consistency of data that has been loaded into BI and is available and used productively there. You use the scenarios that are described below to validate the loaded data. Data reconciliation is based on a comparison of the data loaded into BI and the application data in the source system. You can access the data in the source system directly to perform this comparison.

The term *productive DataSource* is used for DataSources that are used for data transfer in the productive operation of BI. The term *data reconciliation DataSource* is used for DataSources that are used as a reference for accessing the application data in the source directly and therefore allow you to draw comparisons to the source data.

You can use the process for transaction data. Limitations apply when you use the process for master data because, in this case, you cannot total key figures, for example.

Model

The following figure shows the data model for reconciling application data and loaded data in the data flow with transformation. The data model can also be based on 3.x objects (data flow with transfer rules).



The productive DataSource uses data transfer to deliver the data that is to be validated to BI. The transformation connects the DataSource fields with the InfoObject of a DataStore object that has been created for data reconciliation, by means of a direct assignment. The data reconciliation DataSource allows a VirtualProvider direct access to the application data. In a MultiProvider, the data from the DataStore object is combined with the data that has been read directly. In a query that is defined on the basis of a MultiProvider, the loaded data can be compared with the application data in the source system.

In order to automate data reconciliation, we recommend that you define exceptions in the query that proactively signal that differences exist between the productive data in BI and the reconciliation data in the source. You can use information broadcasting to distribute the results of data reconciliation by e-mail, for example.

Modeling Aspects

Data reconciliation for DataSources allows you to check the integrity of the loaded data by, for example, comparing the totals of a key figure in the DataStore object with the corresponding totals that the VirtualProvider accesses directly in the source system.

In addition, you can use the extractor or extractor error interpretation to identify potential errors in the data processing. This function is available if the data reconciliation DataSource uses a different extraction module to the productive DataSource.

We recommend that you keep the volume of data transferred as small as possible because the data reconciliation DataSource accesses the data in the source system directly. This is best performed using a data reconciliation DataSource delivered by BI Content or a generic DataSource using function modules because this allows you to implement an aggregation logic. For mass data, you generally need to aggregate the data or make appropriate selections during extraction.

The data reconciliation DataSource has to provide selection fields that allow the same set of data to be extracted as the productive DataSource.

Selecting the DataSource for Data Reconciliation

Different DataSources can take on the function of a data reconciliation DataSource. The DataSources that can be used in your data reconciliation scenario are explained below.

BI Content DataSources for Data Reconciliation and Recommendations from BI Content for Data Reconciliation

Use the following process to validate your data:

- If a data reconciliation DataSource is specified for a productive DataSource in the BI Content documentation.

  You can see that a DataSource of this type is delivered with BI Content if the documentation in the *Technical Data* table contains an appropriate entry in row *Checkable*.

- If the DataSource documentation contains instructions on building a data reconciliation scenario.

Special DataSources for data reconciliation can be delivered in systems that have PI Basis Release 2005.1 or higher or 4.6C source systems PI 2004.1 SP10.

If the BI Content documentation does not include a reference to a delivered data reconciliation scenario, the decision as to which data reconciliation DataSource you use depends on the properties of the data that is to be compared.

Generic DataSource for Database View or InfoSet

Use this process:

- If BI Content does not deliver a data reconciliation DataSource and the documentation for the productive BI Content DataSource does not include instructions on building a data reconciliation scenario.
- If the data that the productive DataSource supplies is made available in a database table or can be extracted using an InfoSet.
- If you can use selections to significantly limit the volume of data that is to be extracted and transferred.

This process is particularly appropriate for calculated key figures.

If you have created a suitable database view or InfoSet, create a corresponding generic DataSource in the source system in transaction SBIW *Generic DataSources* → *Maintain Generic DataSource*.

Generic DataSource for Function Module

Use this process:

- If BI Content does not deliver a data reconciliation DataSource and the documentation for the productive BI Content DataSource does not include instructions on building a data reconciliation scenario.
- If the data is not available in a database table or cannot be extracted using an InfoSet.
- If you can supply equivalent data for data reconciliation, despite the complex extraction logic of the productive DataSource.

You can reproduce a complex extraction logic using a generic DataSource that extracts data using a customer-defined function module. This allows you to stage data that is equivalent to the productive DataSource, without using the same extraction module as the productive DataSource. In addition, you can use aggregation to reduce the volume of data that is to be transferred.

Note that the extraction logic of the data reconciliation DataSource is prone to errors if the extraction logic of the productive DataSource is complex. Errors in the extraction logic of the data reconciliation DataSource lead to errors in the data reconciliation. We recommend that only experienced developers use this scenario.

Productive DataSource with Direct Access

Use this process:

- If none of the processes described above are possible.
- If the productive DataSource allows direct access.

Since the runtime largely depends on the volume of data that has to be read by the database and transferred, the prerequisite for using this process is that you have set meaningful selections in order to keep the volume of data that is to be transferred small.

During data reconciliation, the data loaded into BI by means of delta transfer is compared with the data in the source system that the extractor accesses directly. Because the same extractor is used for loading and direct access, this process does not allow you to identify potential systematic errors in the logic of the extractor. Errors in processing the delta requests can be identified.

Prerequisites for Performing Data Reconciliation

You have to be able to use suitable selections (time intervals, for example) or pre-aggregation to restrict the scope of the data that you are going to compare so that it can be accessed directly by the VirtualProvider.

In addition, you have to ensure that the selection conditions for the productive DataSource and the data reconciliation DataSource filter the same data range.

Process Flow

1. Create the object model for data reconciliation according to the requirements of your scenario.
2. Load data from the productive DataSource into the DataStore object using suitable selection conditions.
3. Make sure that there is no unloaded data in the delta queue or in the application for the productive DataSource when the check is performed. The application for validating the data is either stopped or the data that is to be reconciled is limited by means of selections (for example, by creating and using time stamps for the data records).
4. Check the data in the query.
5. If you find inconsistencies, proceed as follows:
6. Check whether all the data was loaded from the source system. Load the data that has not yet been loaded into BI, if applicable, and perform reconciliation again.

If the loaded data is not complete, start a repair request.

If the loaded data is complete but not correct, reinitialize the delta or contact SAP.

Example

Information about application-specific scenarios for performing a data reconciliation is available in the How-To-Guide *How to… Reconcile Data Between SAP Source Systems and SAP NetWeaver BI* in SDN at **`http://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uuid/7a5ee147-0501-0010-0a9d-f7abcba36b14`**.

4.3.1.2.8 Delta Process

Definition

The delta process is a feature of the extractor and specifies how data is to be transferred. As a DataSource attribute, it specifies how the DataSource data is passed on to the data target. From this you can derive, for example, for which data a DataSource is suited, and how the update and serialization are to be carried out.

Use

The type of delta process affects the update into a data target. When you update data in an ODS object, you need to serialize it so that you can also overwrite it. According to the delta process, the system decides whether it is necessary to serialize by request or by data packet.

Structure

There are various delta processes for SAP source systems:

1. Forming deltas with after, before and reverse images that are updated directly in the delta queue; an after image shows the status after the change, a before image the status before the change with a negative sign and the reverse image also shows the negative sign next to the record while indicating it for deletion. This serializes the delta packets. The delta process controls whether adding or overwriting is permitted. In this case, adding and overwriting are permitted. This process supports an update in an ODS object as well as in an InfoCube. (technical name of the delta process in the system): ABR)
2. The extractor delivers additive deltas that are serialized by request. This serialization is necessary since the extractor within a request delivers each key once, and otherwise changes in the non-key fields are not copied over correctly. It only supports the addition of fields. It supports an update in an ODS object as well as in an InfoCube. This delta process is used by LIS DataSources. (technical name of the delta process in the system): ADD)
3. Forming deltas with after image, which are updated directly in the delta queue. This serializes data by packet since the same key can be copied more than once within a request. It does not support the direct update of data in an InfoCube. An ODS object must always be in operation when you update data in an InfoCube. For numeric key figures, for example, this process only supports overwriting and not adding, otherwise incorrect results would come about. It is used in FI-AP/AR for transferring line items, while the variation of the process, where the extractor can also send records with the deletion flag, is used in this capacity in BBP. (technical name of the delta process in the system): AIM/AIMD)

Integration

The field 0RECORDMODE determines whether the records are added to or overwritten. It determines how a record is updated in the delta process: A blank character signifies an after image, ‘X’ a before image, ‘D’ deletes the record and ‘R’ means a reverse image.

When you are loading flat files you have to select a suitable delta process from the transfer structure maintenance, this ensures that you use the correct type of update. You can find additional information under InfoSources with Flexible Updating of Flat Files [Page 288].

4.3.1.2.9 unctions in the SAP Source System

Use

The BI Service API (SAPI) is a technology package in the SAP source system that enables the close integration of data transfer from SAP source systems into a BI system.

The SAPI allows you to

- make SAP application extractors available as a basis for data transfer into BI
- carry out generic data extraction
- use intelligent delta processes
- access data in the source system directly from BI (VirtualProvider support)

With transaction SBIW, the SAPI provides an implementation guide in the SAP source system that includes the activities necessary for data extraction and data transfer from an SAP source system into BI.

Irrespective of the type of SAP source system, Customizing for extractors comprises activities that belong to the scope of SAPI:

- general settings for data transfer from a source system into BI
- the option of installing BI Content delivered by SAP
- the option of maintaining generic DataSources
- the option of postprocessing the application component hierarchy and DataSources on a source system level

In addition to the activities that are part of the scope of SAPI, Customizing for extractors for OLTP and further SAP source systems may contain source-system specific settings for application-specific DataSources.

Features

General Settings

General settings include the following activities:

- Maintaining control parameters for data transfer [Page 536]
- Restricting authorizations for extraction [Page 538]
- Monitoring the delta queue [Page 538]

Installing BI Content Delivered by SAP

DataSources delivered with the BI Content by SAP and those delivered by partners appear in a delivery version (D version). If you want to use a partner or BI Content DataSource to transfer data from a source system into BI, you need to transfer this DataSource from the D into the active (A) version.

In the source system, the DataSources are assigned to specific application components. If you want to display the DataSources in BI in the DataSource tree of the Data Warehousing Workbench according to this application component hierarchy, you need to transfer them from the D version into the A version in the source system.

Transferring data from an OLTP system or other SAP source systems

Note: You need to make settings for some BI Content DataSources before you can transfer data into BI. These settings are listed in transaction SBIW in the *Settings for Application-Specific DataSources* section. You can only find this section in those SAP source systems for which it is relevant.

The following activities are associated with installing BI Content:

- Transferring application component hierarchies [Page 540]
- Installing Business Content DataSources [Page 540]

Generic DataSources

Regardless of the specific application, you can use generic data extraction to extract data from any transparent tables, database views or SAP Query functional areas. You do not need to program in ABAP. You can also use function modules for generic data extraction.

In this way, you can use your own DataSources for transaction data, master data attributes or texts. The data for such DataSources is read generically and then transferred into BI.

Generic DataSources allow you to extract data which cannot be supplied to BI either with the DataSources delivered with BI Content or with customer-defined DataSources of the application.

For more information, see Maintaining Generic DataSources [Page 541].

Postprocessing DataSources

You can adapt existing DataSources to suit your requirements as well as edit the application component hierarchy for the DataSources.

For more information, see Editing DataSources and Application Component Hierarchies [Page 550].

4.3.1.2.9.1 Maintaining Control Parameters for Data Transfer

Procedure

Maintain entries for the following fields:

1. Source system

   Enter the logical system for your source client and assign a control parameter to it.

For information about source clients, see the source system under *Tools → Administration → Management → Client Management → Client Maintenance*.

2. Maximum size of the data package

   When you transfer data into BI, the individual data records are sent to BI in packages of variable size. You use this parameter to control the typical size of a data package of this type.

   If you do not maintain an entry, the data is transferred with the default setting of 10,000 kBytes per data package. However, the required memory depends not only on the data package size setting, but also on the width of the transfer structure, the required memory of the affected extractor, and, for large data packages, the number of data records in the package.

3. Maximum number of rows in a data package

   For large data packages, the required memory mainly depends on the number of data records that are transferred with the package. You use this parameter to control the maximum number of data records that you want the data package to contain.

   By default, the system transfers a maximum of 100,000 records per data package.

   The maximum main memory required per data package is approximately 2 X 'Max. Rows' X 1000 bytes.

4. Frequency

   By specifying a frequency you determine the number of data IDocs after which an info IDoc is to be sent. In other words, how many data IDocs are described by a single info IDoc.

   The frequency is set to 1 by default. This means that an info IDoc follows after each data IDoc. You should choose a frequency between 5 and 10, but not greater than 20.

   The larger the package size of a data IDoc, the lower you should set the frequency. As a result, you get information about the data load status during the data upload at relatively short intervals.

   In the BI monitor, you can see from each info IDoc whether the load process was successful. If this is the case for all data IDocs described in an info IDoc, the traffic light in the monitor is green. Info IDocs contain information about whether the data IDocs were correctly uploaded.

5. Maximum number of parallel processes for the data transfer

   An entry in this field is only required as of Release 3.1I.

   Enter a value greater than 0. The maximum number of parallel processes is set to 2 by default. The optimal choice of the parameter depends on the configuration of the application server that you are using for the data transfer.

6. Target system of a batch job

   Enter the name of the application server on which you want to process the extraction job.

   To get the name of the application server, choose *Tools → Administration → Monitor → System Monitoring → Server*. The *Host* column displays the name of the application server.

7. Maximum number of data packages in a delta request

   You use this parameter to set the maximum number of data packages in a delta request or in the repeat of a delta request (repair).

Only use this parameter if you are expecting delta requests with a very large volume of data. In this case, you allow more than 1000 data packages to be generated in a request, while retaining an appropriate data package size.

As before, there are no limits for initials values or the value 0. A limit is only applied if you have a value that is greater than 0. However, for consistency reasons this number is not always strictly adhered to. Depending on the extent to which the data in the qRFC queue is compressed, the actual limit can deviate by up to 100 from the specified value.

4.3.1.2.9.2 Restricting Authorizations for Extraction

Use

You use this function to exclude DataSources from the extraction. Data that is stored in these DataSources is not transferred into BI.

Use this function to exclude DataSources from the extraction for individual BI systems. If you want to exclude a DataSource from the extraction for all connected BI systems, in the post-processing of DataSources, choose **editing DataSources and application component hierarchies [Page 550]** and delete the DataSource.

Procedure

1. Choose *New Entries*.
2. Choose the DataSource that you want to exclude from the extraction.
3. Choose the BI system into which you no longer want data from this DataSource to be extracted.
4. In the *Extr. Off* field, specify that the DataSource is to be excluded from the extraction.
5. Save your entries and specify a transport request.

4.3.1.2.9.3 Delta Queue Check

Use

The delta queue is a data store in the source system into which data records are written automatically. The data records are written to the delta queue either using an update process in the source system (for example with FI documents) or are extracted using a function module when data is requested from BI (for example, LIS extraction prior to BW 2.0).

With a delta request, the data records are transferred into BI from the scheduler.

The data is stored in compressed form in the delta queue. It can be requested from several BI systems. The delta queue is also repeat enabled; it stores the data from the last extraction process. The repeat mode of the delta queue is specific to the target system.

If the extraction structure of a DataSource is changed after data is written into the delta queue but before the queue data is read (for example, when you upgrade), you can tell which structure in the delta queue the data was written to from the data itself. The queue monitor

contains fields that were not filled before but are now filled and/or fields that were filled before but are no longer filled.

You use this function to check the delta queue.

Features

The status symbol shows whether an update into a delta queue is activated for a particular DataSource. The delta queue is active if the status symbol is green; it is filled with data records when there is an update process or data request from BI. The delta method has to be initialized successfully in the scheduler in BI before a delta update can take place.

You can carry out the following activities:

- Display data records
- Display the current status of the delta-relevant field
- Refresh
- Delete the queue
- Delete queue data

Activities

Displaying Data Records

1. To check the amount and type of data in the delta queue, select the delta queue and choose *Display Data Records.*
2. A dialog box appears in which you can specify how you want to display the data records.
    a. You can select the data packages that contain the data records you want to see.
    b. You can display specific data records in the data package.
    c. You can simulate the extraction parameters to select how you want to display the data records.
3. To display the data records, choose *Execute.*

Displaying Current Status of Delta-Relevant Field

For DataSources that support generic deltas, you can display the current value of the delta-relevant field in the delta queue. In the *Status* column, choose *Detail*. The value displayed includes the largest value for the last extraction with reference to the delta-relevant field. It is the lower limit for the next extraction.

Refreshing

If you select refresh,

- newly activated delta queues are displayed
- new data records that have been written to the delta queue are displayed
- data records that have been deleted by the time the system reads the data records are not displayed

Deleting Queue Data

To delete the data in a delta queue for a DataSource, select the delta queue and in the context menu, choose *Delete Data.*

If you delete data from the delta queue, you do not have to reinitialize the delta method to write the DataSource data records into the delta queue.

Note that data is also deleted that has not yet been read from the delta queue. As a result, any existing delta update is invalidated. Only use this function when you are sure that you want to delete all queue data.

Deleting Queues

You can delete the entire queue by choosing *Queue* → *Delete Queue*. You need to reinitialize the delta method before you can write data records for the related DataSource into the delta queue.

4.3.1.2.9.4 Installing Application Component Hierarchies

You use this function to install and activate application component hierarchies delivered by SAP or by partners.

After the DataSources are replicated in BI, this application component hierarchy is displayed with the transferred DataSources in the source system view of the Data Warehousing Workbench – Modeling. In BI, choose the DataSource overview from the context menu (right-mouse click) for the source system.

If you activate the BI Content application component hierarchy, the active customer version is overwritten when you install the BI Content version.

For information about changing the installed application component hierarchy, see Editing DataSources and Application Component Hierarchies [Page 550].

4.3.1.2.9.5 Installing BI Content DataSources

Use

You use this function to transfer and activate DataSources delivered with BI Content and, where applicable, partner DataSources delivered in their own namespaces. After installing BI Content DataSources you can extract data from all the active DataSources that you have replicated in BI and transfer this data to all connected BI systems.

Activities

The *Install DataSources from BI Content* screen displays the DataSources in an overview tree. This tree is structured in accordance with the application components assigned to you.

1. In the application component hierarchy, select the nodes for which you want to install DataSources in the active version. To do this, position the cursor on the node and choose *Highlight Subtree.*

   The DataSources and subtrees below the node are selected.

2. Choose *Select Delta*.

DataSources where the system found differences between the active and the delivered version (due to changes to the extractor, for example) are highlighted in yellow.

3. To analyze the differences between active and delivered versions of a particular DataSource, select the DataSource and choose *Version Comparison.* The application log contains further information about the version comparison.
4. To transfer a DataSource from the delivery version to the active version, select it in the overview tree by choosing *Highlight Subtree* and choose *Transfer DataSources.*

   If an error occurs, the error log appears.

   Regardless of whether data has been successfully transferred into the active version, you can call the log by choosing *Display Log.*

With a metadata upload (when you replicate DataSources in BI), the active version of the DataSource is made known to BI.

When you activate BI Content DataSources, the system overwrites the active customer version with the SAP version.

You can only search for DataSources or other nodes in expanded nodes.

For information about changing the installed DataSources, see Editing DataSources and Application Components [Page 550].

4.3.1.2.9.6 Maintaining Generic DataSources

Use

Regardless of the application, you can create and maintain generic DataSources for transaction data, master data attributes or texts from any transparent table, database view or SAP Query InfoSet, or using a function module. This allows you to extract data generically.

Procedure

Creating Generic DataSources

1. Select the DataSource type and specify a technical name.
2. Choose *Create.*

   The screen for creating a generic DataSource appears.
3. Choose the application component to which you want to assign the DataSource.
4. Enter the descriptive texts. You can choose any text.
5. Select the datasets from which you want to fill the generic DataSource.
   a. Choose *Extraction from View* if you want to extract data from a transparent table or a database view. Enter the name of the table or the database view.

      After you generate the DataSource, you have a DataSource with an extraction structure that corresponds to the database view or transparent table.

      For more information about creating and maintaining database views and tables, see the *ABAP Dictionary Documentation*.

b. Choose *Extraction from Query* if you want to use a SAP Query InfoSet as the data source. Select the required InfoSet from the InfoSet catalog.

Notes on Extraction Using SAP Query [Page 549]

After you generate the DataSource, you have a DataSource with an extraction structure that corresponds to the InfoSet.

For more information about maintaining the InfoSet, see the *System Administration* documentation.

c. Choose *Extraction Using FM* if you want to extract data using a function module. Enter the function module and extraction structure.

The data must be transferred by the function module in an interface table E_T_DATA.

Interface Description and Extraction Process [Page 546]

For information about the function library, see the *ABAP Workbench: Tools* documentation.

d. With texts you also have the option of extracting from fixed values for domains.

6. Maintain the settings for delta transfer, as required.
7. Choose *Save*.

When performing extraction, note SAP Query: Assigning to a User Group [Page 549].

Note when extracting from a transparent table or view:

If the extraction structure contains a key figure field that references a unit of measure or a currency unit field, this unit field has to be included in the same extraction structure as the key figure field.

A screen appears on which you can edit the fields of the extraction structure.

8. Edit the DataSource:
   - Selection

     When you schedule a data request in the BI scheduler, you can enter the selection criteria for the data transfer. For example, you can determine that data requests are only to apply to data from the previous month.

     If you set the *Selection* indicator for a field within the extraction structure, the data for this field is transferred in correspondence with the selection criteria in the scheduler.

   - Hide field

     You set this indicator to exclude an extraction structure field from the data transfer. The field is no longer available in BI when you set the transfer rules or generate the transfer structure.

   - Inversion

     Reverse postings are possible for customer-defined key figures. Therefore inversion is only active for certain transaction data DataSources. These include DataSources that have a field that is marked as an inversion field, for example, the **update mode** field in DataSource **0FI_AP_3**. If this field has a value, the data records are interpreted as reverse records in BI.

If you want to carry out a reverse posting for a customer-defined field (key figure), set the *Inversion* indicator. The value of the key figure is transferred to BI in inverted form (multiplied by –1).

- Field only known in exit

  You can enhance data by extending the extraction structure for a DataSource by adding fields in append structures.

  The *Field Only Known in Exit* indicator is set for the fields of an append structure; by default these fields are not passed to the extractor from the field list and selection table.

  Deselect the *Field Only Known in Exit* indicator to enable the Service API to pass on the append structure field to the extractor together with the fields of the delivered extract structures in the field list and in the selection table.

9. Choose *DataSource → Generate.*

The DataSource is saved in the source system.

Maintaining Generic DataSources

- Change DataSource

  To change a generic DataSource, in the initial screen of DataSource maintenance, enter the name of the DataSource and choose *Change.*

  You can change the assignment of a DataSource to an application component or change the texts of a DataSource. Double-click on the name of the table, view, InfoSet or extraction structure to get to the appropriate maintenance screen. Here you make the changes to add new fields. You can also completely swap transparent tables and database views, though this is not possible with InfoSets. Return to DataSource maintenance and choose *Create*. The screen for editing a DataSource appears. To save the DataSource in the SAP source system, choose *DataSource → Generate.*

  If you want to test extraction in the source system independently of a BI system, choose *DataSource → Test Extraction.*

- Delta DataSource

  On the *Change Generic DataSource* screen, you can delete any DataSources that are no longer relevant. If you are extracting data from an InfoSet, delete the corresponding query. If you want to delete a DataSource, make sure it is not connected to a BI system.

For more information about extraction using SAP Query, see Extraction Using SAP Query [Page 548].

4.3.1.2.9.6.1 Delta Transfer to BI

The following update modes are available in BI:

- Full update

  A full update requests all data that meets the selection criteria you set in the scheduler.

- Delta update

  A delta update only requests data that has appeared in the source system since the last load.

- Initializing the delta process

You need to initialize a delta process before you can execute it. The initialization selections are copied to load the delta records.

With large volumes of data, you can only ensure a performance-optimized extraction from the source system if you use a delta process.

In the maintenance of the generic DataSource, you can set up a delta for master data attributes and texts. You can also set up a generic delta using a (delta-relevant) field with a monotonically increasing value.

Setting Up an ALE Delta for Master Data Attributes or Texts

Master data attributes or texts for which you want to use a delta transfer have to fulfill two prerequisites:

1. Data must be extracted generically using a transparent table or database view.
2. A change document object must be available that can update the complete key of the table (or view) used for data extraction in combination with one of the tables on which the change document object is based.

The required control entries are delivered for the most important master data attributes and texts. By including a maintenance interface for control entries in the maintenance of generic DataSources or InfoSources, you can use the delta transfer for other master data attributes or texts.

To generate the control entry for master data attributes or texts that is required for BI, proceed as follows:

1. For an attribute or text DataSource, choose *DataSource* → *ALE Delta*.
2. Enter the table and the change document object that you want to use as a basis for the delta transfer.

   An input help for the *Table Name* field searches all possible tables for a suitable key.
3. Confirm your entries.

   If you have entered a usable combination of table and change document object, the extraction structure fields are listed in the table below. The status in the first column shows whether changing the master data in this field causes the system to transfer the delta record.
4. Save the settings to generate the required control entry.

   Delta transfer is now possible for master data and texts.

After the DataSource has been generated, you can see this on the *DataSource: Edit Customer Version* screen; the *Delta Update* field is selected.

You need two separate entries if you want to transfer delta records for texts and master data attributes.

Generic Delta

If a field exists in the extraction structure of a DataSource and the field contains values that increase monotonically over time, you can define delta capability for this DataSource. If a delta-relevant field of this type exists in the extraction structure (a timestamp, for example), the system determines the data volume transferred in the delta mode by comparing the maximum value transferred with the last load with the amount of data that has since entered the system. Only the new data is transferred.

To get the delta, generic delta management translates the update mode into a selection criterion. The selections of the request are enhanced with an interval for the delta-relevant field. The lower limit of the interval is taken from the previous extraction. The upper limit is taken from the current value, for example, the timestamp at the time of extraction. You use security intervals to ensure that all data is taken into account during extraction (see below). After the data request is transferred to the extractor and the data is extracted, the extractor informs generic delta management that the pointer can be set to the upper limit of the previously determined interval.

The delta for generic DataSources cannot be used with a BI system release prior to 3.0. In older SAP BW releases, the system does not replicate DataSources for master data and texts that were delta-enabled using the delta for generic DataSources.

Determining Generic Delta for a DataSource

1. Choose *Generic Delta*.
2. In the dialog box that appears, specify the delta-determining field and the type of this field.
3. Maintain the settings for the generic delta:
   a. Enter a security interval.

      The purpose of a security interval is to make the system extract records with the next extraction that appear during the extraction process but which have not yet been extracted (because, for example, they have not been saved).

      You can add a security interval to the upper limit or lower limit of the interval.

      You should only specify a security interval for the lower limit if the delta process produces a new status for the changed records (when the status is overwritten in BI). In this case, duplicate data records that may arise with a security interval of this type have no affect in BI.

   b. Choose the delta type for the data that you want to extract.

      You use the delta type to determine how the extracted data is interpreted in BI and the data targets to which it can be updated.

      With the *Additive Delta* delta type, the record to be loaded for cumulative key figures only returns the change to the respective key figure. The extracted data is added into BI. DataSources with this delta type can write data to DataStore objects and InfoCubes.

      With the *New Status for Changed Records* delta type, each record to be loaded returns the new status for all key figures and characteristics. The values in BI are overwritten. DataSources with this delta type can write data to DataStore objects and master data tables.

   c. Specify whether the DataSource supports real-time data acquisition.
4. Save your entries.

   Delta transfer is now possible for this DataSource.

After the DataSource has been generated, you can see this on the *DataSource: Edit Customer Version* screen; the *Delta Update* field is selected.

In systems as of Basis Release 4.0B, you can display the current value of the delta-relevant field in the delta queue.

Example of Determining Selection Intervals with Generic Delta

Security interval upper limit

The delta-relevant field is a timestamp.

The timestamp that was read last is 12:00:00. Delta extraction begins at 12:30:00. The security interval for the upper limit is 120 seconds. The selection interval for the delta request is: 12:00:00 to 12:28:00. When the extraction is finished, the pointer is set to 12:28:00.

Security interval lower limit

The delta-relevant field is a timestamp. After images are transferred. In BI the record is overwritten with the post-change status, for example, for master data. Any duplicate data records do not affect BI.

The last read timestamp is 12:28:00. Delta extraction begins at 13:00. The security interval for the lower limit is 180 seconds. The selection interval for the delta request is: 12:25:00 to 13:00:00. When the extraction is finished, the pointer is set to 13:00:00.

4.3.1.2.9.6.2 Function Module: Interface Description and Procedure

A description of the interface for a function module that is used for generic data extraction:

Importing Parameter

I_DSOURCE type SRSC_S_IF_SIMPLE-DSOURCE DataSource

I_INITFLAG type SRSC_S_IF_SIMPLE-INITFLAG Initialization call

I_MAXSIZE type SRSC_S_IF_SIMPLE-MAXSIZE package size

I_REQUNR type SRSC_S_IF_SIMPLE-REQUNR request number

Tables

I_T_SELECT type SRSC_S_IF_SIMPLE-T_SELECT

I_T_FIELDS type SRSC_S_IF_SIMPLE-T_SELECT

E_T_DATA

Exceptions

NO_MORE_DATA

ERROR_PASSED_TO_MESS_HANDLER

Details on Individual Parameters

- I_INITFLAG

  This parameter is set to ‘X’ when the function module is called up for the first time, then to ‘ ‘.

- I_MAXSIZE

  This parameter contains the number of lines expected within a read call.

Extraction Process

The function module is called up again and again during an extraction process:

1. Initialization call:

   Only the request parameters are transferred to the module here. The module is as yet unable to transfer data.

2. First read call:

   The extractor returns the data typified with the extract structure in an interface table. The number of rows the system expected is determined in the request parameter (I_MAXSIZE).

3. Second read call:

   The extractor returns the data enclosed within the first data package in a separate package with I_MAXSIZE rows.

4. The system calls up the function module again and again until the module returns the exception NO_MORE_DATA. Data cannot be transferred in the call in which this exception is called up.

Example

An example of a function module that meets these demands is RSAX_BIW_GET_DATA_SIMPLE. A simple way of creating a syntactically correct module is to copy it into its own function group and then to cope the rows of the top-include of function group RSAX (LRSAXTOP) into the top-include of its own function group. Afterwards, the copied function module must be adjusted to the individual requests.

4.3.1.2.9.6.3 Testing Extraction

Use

You can use this function to test extraction from DataSources that were created using the maintenance for the generic DataSource. After the test extraction, you can display the extracted data and the associated logs.

Procedure

1. Choose **DataSource** → *Test Extraction*.

   A screen appears in which you can set parameters and selections for the test extraction.

2. Enter a request no. for the test extraction via a function module.

3. Enter how many data records are to be read with each extractor call.

4. The extractor is called up by the Service API until no more data is available. In the *Display Extr. Calls* field, you can specify the maximum number of times the extractor is to be called. This enables you to restrict the no. of data packages when testing the extraction. With a real extraction, the system transfers data packages until is no longer able to find any more data.

5. Depending on the definition of the DataSource, you can test the extraction in various update modes.

For DataSources that support the delta method, you can also test deltas and repeats as well as the full update.

The modes delta and repeat are only available for testing when the extractor supports a mode in which the system reads the data but does not modify the delta management status tables.

- To avoid errors in BW, the timestamp or pointer that was set in delta management must not be changed during testing.
- Before you are able to test the extraction in a delta mode in the source system, you need to have carried out an initialization of the delta method or a simulation of such an initialization for this DataSource.

You can test the transfer of an opening balance for non-cumulative values.

6. Specify selections for the test extraction.

   Only those extract structures fields you have selected in DataSource maintenance can be selected.

   To enter several selections for a field, insert new rows for this field into the selection table.

7. Choose whether you want to execute the test extraction in debug mode or by tracing an authorization trace.

   If you test the extraction in the debug mode, a breakpoint is set just before the extractor initialization call.

   For information on the debugger, see **ABAP Workbench: Tools**.

   If you set an authorization trace, you can call it after the test by choosing *Display Trace.*

8. Start the extraction.

Result

If the extraction was successful, a message appears that specifies the number of extracted records. The buttons *Display List, Display Log* and *Display Trace* (optional) appear on the screen. You can use *Display List* to display the data packages. By double-clicking on the number of records for a data package, you get to a display of the data records. Choose *Display Log* to display the application log.

4.3.1.2.9.6.4 Extraction Using SAP Query

SAP Query is a comprehensive tool for defining reports. It uses many different forms of reporting. It allows users to define and execute their own evaluations of data in the SAP system without requiring ABAP programming know-how.

To define the structure of evaluations, you enter texts in SAP Query and select fields and options. InfoSets and functional groups allow you to easily select the relevant fields.

An InfoSet is a special view of a set of data (logical database, table join, table, sequential file). It serves as the data source for SAP Query. An InfoSet determines which tables or fields of these tables are referenced in an evaluation. InfoSets are usually based on logical databases.

The maintenance of InfoSets is one component of SAP Query. When an InfoSet is created, a data source is selected in an application system. Since a data source can have a large

number of fields, fields can be combined into logical units; the functional groups. Functional groups are groups of several fields that form a logical unit within an InfoSet. Any fields that you want to use in an extraction structure have to be assigned to a functional group. In generic data extraction using an InfoSet, all the fields of all functional groups for this InfoSet are available.

The relevance of SAP Query to BI lies in the definition of the extraction structure by selecting fields of a logical database, a table join or other datasets in an InfoSet. This allows you to use generic data extraction for master or transaction data from any InfoSet. A query is generated for an InfoSet. The query gets the data and transfers it to the generic extractor.

InfoSets represent an additional, easily manageable data source for generic data extraction. They allow you to use logical databases from all SAP applications, table joins, and further datasets as data sources for BI. For more information about SAP Query, and InfoSets in particular, see the SAP Query documentation -> System Administration.

In the following section, the terms SAP Query and InfoSet are used independently of the source system release. Depending on the source system release, SAP Query is the same as an ABAP Query or ABAP/4 query. The InfoSet is also called a functional area in some source system releases.

4.3.1.2.9.6.4.1 Notes on Extraction Using SAP Query

Client Dependency

InfoSets are only available if you have created them globally, independent of a client. You set this global area in the initial screen of InfoSet maintenance under *Environment → Work Areas.*

Size Limits When Extracting Data Using SAP Query InfoSets

If you are using an InfoSet to extract data, the system first collects all data in the main memory. The data is transferred to the BI system in packages using the Service API interface. The size of the main memory is therefore important with this type of extraction. It is suitable for limited datasets only.

As of SAP Web Application Server 6.10, you can extract mass data using certain InfoSets (tables or table joins).

See also:

Extraction Using SAP Query [Page 548]

4.3.1.2.9.6.4.2 SAP Query: Assignment to a User Group

If you want to extract your data from an InfoSet, the InfoSet must be assigned to a user group before the DataSource can be generated. This is necessary as the extraction is processed from an InfoSet using a query that comprises all fields of the InfoSet. In turn, this query can only be generated when the InfoSet is assigned to a user group.

Releases up to 3.1I

In releases up to 3.1I, a screen appears in which you have to specify a user group as well as a query name. The user group must be specified using the value help. In other words, it must already have been created. You can get more information about creating user groups in the SAP Query documentation, in the section *System Management → Functions for Managing User Groups*.

A separate query is required for an InfoSet each time it is used in a DataSource. For this reason, enter a query name that was previously not in the system.

The query is generated after you confirm your entries.

Releases from 4.0A

In releases as of 4.0A, the InfoSet for the extract structure of the new DataSource is automatically assigned to the pre-finished system user group. A query is automatically generated by the system.

4.3.1.2.9.7 Editing DataSources and Application Component Hierarchies

Use

To adapt existing DataSources to your requirements, you can edit them in this step before transporting them from a test system into a productive system.

In this step you can also postprocess the application component hierarchy.

Procedure

DataSource

Transporting DataSources

Select the DataSources that you want to transport from the test system into the productive system and choose *Transport*. Specify a development class and a transport request so that the DataSources can be transported.

Maintaining DataSources

To maintain a DataSource, select it and choose *Maintain DataSource*. The following editing options are available:

- Selection

  When you schedule a data request in the BI scheduler, you can enter the selection criteria for the data transfer. For example, you can determine that data requests are only to apply to data from the previous month.

  If you set the *Selection* indicator for a field within the extraction structure, the data for this field is transferred in correspondence with the selection criteria in the scheduler.

- Hide field

  You set this indicator to exclude an extraction structure field from the data transfer. The field is no longer available in BI when you set the transfer rules or generate the transfer structure.

- Inversion

Reverse postings are possible for customer-defined key figures. Therefore inversion is only active for certain transaction data DataSources. These include DataSources that have a field that is marked as an inversion field, for example, the **update mode** field in DataSource **0FI_AP_3**. If this field has a value, the data records are interpreted as reverse records in BI.

Set the *Inversion* indicator if you want to carry out a reverse posting for a customer-defined field (key figure). The value of the key figure is transferred to BI in inverted form (multiplied by –1).

- Field only known in exit

  You can enhance data by extending the extraction structure for a DataSource by adding fields in append structures.

  The *Field Only Known in Exit* indicator is set for the fields of an append structure; by default these fields are not passed to the extractor from the field list and selection table.

  Deselect the *Field Only Known in Exit* indicator to enable the BI Service API to pass on the append structure field to the extractor together with the fields of the delivered extract structures in the field list and in the selection table.

Enhancing the extraction structure

If you want to transfer additional information for an existing DataSource from a source system into BI, you first need to enhance the DataSource extraction structure by adding fields. To do this, create an append structure for the extraction structure (see Adding Append Structures [External]).

1. Choose *Enhance Extr. Str.,* to access field maintenance for the append structure. The name of the append structure is taken from the extraction structure name in the customer namespace.
2. Enter the fields you want to add in the field list, together with their subordinate data elements. You can use all the functions that are available for maintaining fields of tables and structures.
3. Save and activate your append.
4. Go back to the DataSource display and make sure that the *Hide Field* indicator is not selected for the newly added fields.

Function enhancement

To fill the append structure fields with data, you need to create a customer-specific function module. For information about enhancing the SAP standard with customer-specific function modules, see *Enhancing the SAP Standard* in SAP Library.

The SAP enhancement **RSAP0001** is available for enhancing BI DataSources. This enhancement contains the following enhancement components:

Transaction data	exit_saplrsap_001
Attributes, texts	exit_saplrsap_002
Hierarchies	exit_saplrsap_004

For more information, see Enhancing DataSources [Page 552].

As of Release 6.0, the Business Add-In (BAdI) **RSU5_SAPI_BADI** is available. You can display the BAdI documentation in the BAdI definition or BAdI implementation.

Application Component Hierarchy

- To create a same-level or lower-level node for a particular node, place the cursor over this node and choose *Object → Create Node.* You can also create lower-level nodes by choosing *Object → Create Children.*
- To rename, expand, or compress a node, place your cursor over the node and click on the appropriate button.
- To move a node or subtree, select the node you want to move (by positioning the cursor over it and choosing *Select Subtree)*, position the cursor on the node onto which the selected node is to be positioned. Choose *Reassign.*
- If you select a node with the cursor and choose *Set Segment,* this node is displayed with its subnodes. You can go to the higher-level nodes for this subtree using the appropriate links in the row above the subtree.
- If you select a node with the cursor and choose *Position,* the node is displayed in the first row of the view.
- All DataSources for which a valid (assigned) application component could not be found are placed under the node NODESNOTCONNECTED. The node and its subnodes are only built at transaction runtime and are refreshed when the display is saved.

  NODESNOTCONNECTED is not persistently saved to the database and is therefore not transferred in a particular state to other systems when you transport the application component hierarchy.

  Note: Hierarchy nodes created under NODESNOTCONNECTED are lost when you save. After you save, the system only displays those nodes under NODESNOTCONNECTED that were moved to this node with DataSources.

  A DataSource is positioned under an application component X. You transfer a new application component hierarchy from BI Content that does not contain application component X. In this application component, the DataSource is automatically placed under the node NODESNOTCONNECTED.

  Note: Changes to the application component hierarchy only apply until BI Content is installed again.

4.3.1.2.9.7.1 Enhancing DataSources

Use

The SAP enhancement **RSAP0001** is available if you want to fill fields that you have added to the extraction structure of a DataSource as an append structure. This enhancement is made up of the following enhancement components:

Data Type	Enhancement Component
Transaction data	exit_saplrsap_001
Attributes, texts	exit_saplrsap_002
Hierarchies	exit_saplrsap_004

See also:

Changing the SAP Standard [External] → Customer Exits [External]

Prerequisites

You have enhanced the extraction structure of the DataSource with additional fields.

Procedure

Note:

As soon as an SAP enhancement is assigned to one project, it can no longer be copied to another project.

5. In Customizing, choose the extractors (transaction SBIW in source system) *Postprocessing DataSources → Edit DataSources and Application Component Hierarchy*.
6. Highlight the DataSource that you want to enhance and choose *DataSource Function Enhancement*.

   The project management for SAP enhancements screen appears.
7. Specify a name for your enhancement project in the *Project* field.
8. Choose *Project → Create.*

   The *Attribute Enhancement Project <Project Name>* screen appears.

   Note:

   If a project has already been created for the SAP enhancement, use the existing project and continue with step i).
9. Enter a short description for your project.
10. Save the attributes for the project.
11. Choose *Goto → Enhancement Assignment*.
12. In the *Enhancement* field, enter the name of the SAP enhancement that you want to edit, in this case **RSAP0001**.

    You can combine several SAP enhancements in one enhancement project.

    To display the SAP documentation for an SAP enhancement, highlight the SAP enhancement and choose Goto -> Display Documentation.
13. Save your entries.

    If a project already exists for the SAP enhancement, you cannot save your entries. Go back to the initial screen and enter the existing project. Continue with step i).
14. Return to the start screen.
15. Select the C*omponent* subobject.
16. Choose *Change*.

    The system displays the SAP enhancements you have entered with the corresponding components (in this case, function exit).

    To display the documentation for a component, select the component and choose *Goto → Display Documentation*.
17. Select the component (for example, EXIT_SAPLRSAP_001) that you want to edit and choose *Edit → Select*.

    The system displays the function module prepared by the SAP application developer. Use the include program contained in this module to transfer your functionality to the module.

18. Call the include program by double-clicking on it.
    - The ABAP editor appears.
    a. Enter the source text for your function in the editor and save your include program.
    - The system asks whether you want to create an include program.
    a. Confirm that you want to create an include program.
    b. Specify the program attributes and save them.
    c. Choose *Goto* → *Source Code.*

       The ABAP editor appears.
    d. Enter the source text for your function in the editor and save your include program.
19. Return to the start screen.
20. Activate your enhancement project by choosing *Project* → *Activate Project*.

See also:

Creating Additional Projects [External]

Creating Customer-Specific Function Modules [External]

Result

The enhancement is activated and at the runtime of the extractor, the fields that have been added to the DataSource using the append structure are filled with data.

4.3.1.2.10 Functions for DataSource 3.x in Data Flow 3.x

4.3.1.2.10.1 Assigning DataSources 3.x to InfoSources 3.x and Fields to InfoObjects

Use

You carry out the assignment of a DataSource 3.x to an InfoSource in the BI transfer rules maintenance. An InfoSource can have multiple DataSources assigned to it if you want to consolidate data from different sources.

The fields for a DataSource 3.x are assigned to InfoObjects in BI. This assignment takes place in the same way in the transfer rules maintenance.

For BI Content DataSources, the assignment to InfoSources, as well as the assignment of fields to InfoObjects, is delivered by SAP.

4.3.1.2.10.2 Transfer Structure in Data Flow 3.x

Definition

The transfer structure is the structure in which the data is transported from the source system into BI.

It is a selection of DataSource fields from a source system.

Use

The transfer structure provides BI with all the source system information available for a business process.

An InfoSource 3.x in BI needs at least a DataSource 3.x for data extraction. In an SAP source system, DataSource data that logically belongs together is staged in a flat structure, the extraction structure. In the source system, you are able to filter and enhance the extraction structure in order to determine the DataSource fields.

In the transfer structure maintenance in BI, you determine which fields of the DataSource 3.x are to be transferred to BI. When you activate the transfer rules in BI, a transfer structure identical to the one in BI is created in the source system from the DataSource fields.

This data is transferred 1:1 from the transfer structure of the source system into the BI transfer structure, and is then transferred into the BI communication structure using the transfer rules.

A transfer structure always refers to a DataSource from a source system and to an InfoSource in BI.

If you choose *Create Transfer Rules* from the DataSource or the InfoSource in an object tree of the Data Warehousing Workbench, the transfer structure maintenance appears.

4.3.1.3 Transferring Data Using Web Services

Purpose

Data is generally transferred into BI by means of a data request, which is sent from BI to the source system (pull from the scheduler). You can also use Web services if you want the data transfer to be controlled from outside the BI system and sent into the inbound layer of BI, the Persistent Staging Area (PSA). This is a data push into the BI system.

If you are using Web services to transfer data into BI, you can use real-time data acquisition to update the data into BI. Alternatively, you can update data using a standard data transfer process:

- If you access the data frequently and want the data to be refreshed every once an hour to every once a minute, use Real-Time Data Acquisition [Page 630]. The data is first written to the PSA of the BI system. From there, the data is controlled by a background process, or daemon, which runs at frequent regular intervals; the data is updated to a DataStore object and is then immediately available for operational reporting.
- If you do not need to refresh the data in BI on an hourly basis to meet your analysis and reporting requirements, use the standard update. Again, the data is first written to the PSA of the BI system. Process chains control the update and further processing of data.

In SAP NetWeaver 2004s, you generate Web services for data loading when you activate a DataSource defined in the BI system. The Web services provide you with WSDL descriptions, which can be used to send data to BI regardless of the technology used.

The BI server SOAP interface can ensure **guaranteed delivery**, since an XML message is returned to the client upon success as well as failure. If the client receives an error or no message at all (due to connection termination when sending a success message, for example), the client can resend the data.

It is not currently possible to ensure **guaranteed delivery only once**, since there is no match at transaction-ID level. This is required to determine whether a data package was 'inadvertently' resent and should not be updated. If deltas are built using after-images (delta process AIM), the update to a DataStore object can, however, consistently deal with data sent excessively, as long as serialization is guaranteed. The serialization is the task of the client.

Prerequisites

You are familiar with the Web service standards and technology.

See also:

Web Services [External]

Process Flow

Design Time

1. You define the Web service DataSource in BI. When you activate the DataSource, the system generates an RFC-enabled function module for the data transfer, along with a Web service definition, which you can use to generate a client proxy in an external system. For example, you can implement the Web service in ABAP in SAP systems.
2. Depending on how you want to update data to BI, proceed as follows:
   - You specify the dataflow for real-time data acquisition.
     i. After you have defined a DataStore object, you create a transformation with the DataSource as the source and the DataStore object as the target; you also create a corresponding data transfer process for real-time data acquisition.

        You have to use a standard data transfer process to further update data to subsequent InfoProviders.

     ii. In an InfoPackage for real-time data acquisition, you specify the threshold values for the size of the data packages and requests; this information is required to process the sent data.

     iii. In the monitor for real-time data acquisition, you define a background process (daemon) and assign the DataSource (with InfoPackage) and data transfer process to it.

     For more information, see Transferring Transaction Data Using Web Services (RDA) [Page 638].
   - You specify the dataflow for the standard update:

i. After you have defined an InfoProvider, you create a transformation with the DataSource as the source and the InfoProvider as the target; you also create a corresponding (standard) data transfer process. Specify any subsequent InfoProviders, transformations and data transfer processes, as required.

ii. In an InfoPackage, you specify the threshold values for the size of the data packages and requests; this information is required to process the sent data.

Since it is only possible to specify threshold values in an *InfoPackage for Real-Time Data Acquisition*, this type of InfoPackage is also used with the standard update.

As with real-time data acquisition, the PSA request remains open across several load processes. The system automatically closes the PSA request when one of the threshold values defined in the InfoPackage is reached. If you want to update data using a standard data transfer process, it must also be possible to close the PSA request without waiting for the threshold values to be reached. This is controlled in a process chain by the process type *Close Real-Time InfoPackage Request*.

iii. You create a process chain to control data processing in BI.

This process chain starts with process *Close Real-Time InfoPackage Request*; update processes and processes for further processing are included in the process chain.

For more information, see Transferring Transaction Data Using Web Services (Standard) [Page 559].

Runtime

You use the Web service to send data to the PSA of the BI system.

A WSDL description of the Web service, along with a test function to call the Web service, is available in administration for the SOAP runtime (transaction WSADMIN).

If you are using real-time data acquisition and the daemon is running, the daemon controls the regular update of data from the PSA to the DataStore object. The data is activated automatically and is available immediately for analysis and reporting.

If you are using standard update and the process chain is running, the process chain controls when the PSA request is closed and triggers the processes for update and further processing.

4.3.1.3.1 Creating Web Service Source Systems

1. In the source system tree in the Data Warehousing Workbench, choose *Create* in the context menu for *Web Service*.

2. In the *Logical System Name* field, enter a technical name for the source system.
3. Enter a description for the source system.
4. In the *Type and Release* field, enter the type of source from a semantic perspective.

If SAP ships BI Content for a non-SAP source system, a source type and source release are assigned to this content. If you are using the corresponding system, the correct BI Content can only be found if you specify the source type and source release here.

4.3.1.3.2 Creating DataSources for Web Services

Use

In order to transfer data into BI using a Web service, the metadata first has to be available in BI in the form of a DataSource.

Procedure

You are in the DataSource tree in the Data Warehousing Workbench.

1. Select the application components in which the DataSource is to be created and choose *Create DataSource*.
2. In the next screen, enter a technical name for the DataSource, select the type of the DataSource and choose *Copy*.

   The DataSource maintenance screen appears.
3. Go to the *General* tab page.
   a. Enter descriptions for the DataSource (short, medium, long).
   b. If necessary, specify whether the DataSource may potentially deliver duplicate data records within a request.
4. Go to the *Extraction* tab page.

   Define the delta method for the DataSource.

   DataSources for Web services support real-time data acquisition. Direct access to data is not supported.
5. Go to the *Fields* tab page.

   Here you determine the structure of the DataSource either by defining the fields and field properties directly, or by selecting an InfoObject as a *Template InfoObject* and transferring its technical properties for the field in the DataSource. You can modify the properties that you have transferred from the InfoObject further to suit your requirements by changing the entries in the field list.

   Entering InfoObjects here does not equate to assigning them to DataSource fields. Assignments are made in the transformation. When you define the transformation, the system proposes the InfoObjects you entered here as InfoObjects that you might want to assign to a field.
6. Save and activate the DataSource.

7. Go to the *Extraction* tab page.

   The system has generated a function module and a Web service with the DataSource. They are displayed on the *Extraction* tab page. The Web service is released for the SOAP runtime.

8. Copy the technical name of the Web service and choose *Web Service Administration*.

   The administration screen for SOAP runtime appears. You can use the search function to find the Web service. The Web service is displayed in the tree of the *SOAP Application for RFC-Compliant FMs*. Select the Web service and choose *Web Service → WSDL* (Web Service Description Language) to display the WSDL description.

Result

The DataSource is created and is visible in the Data Warehousing Workbench in the application component in the DataSource overview for the Web service source system. When you activate the DataSource, the system generates a PSA table and a transfer program.

Before you can use a Web service to transfer data into BI for the DataSource, create a corresponding InfoPackage (push package). If an InfoPackage is already available for the DataSource, you can test the Web service push in Web service administration.

See also:

Web Services [External]

4.3.1.3.3 Transferring Data Using Web Services (Standard)

Use

If you want data transfer into BI (master data or transaction data) to be controlled externally, as opposed to being requested by BI, and you do not need to refresh data more than once an hour, use the Web service with standard update to transfer data into the BI system.

Procedure

1. Create a Web service DataSource.

   See Creating DataSources for Web Services [Page 558].

2. Implement the Web service in your application.
3. Create a suitable InfoProvider.
4. Create a transformation with the DataSource as the source and the InfoProvider as the target.

   See Creating Transformations [Page 242].

5. Create an InfoPackage for the DataSource for real-time data acquisition.

   See Creating InfoPackages for Real-Time Data Acquisition [Page 647].

PSA requests for Web services remain open across several load processes. When you transfer data using Web services, you use this type of InfoPackage to define the size of the request or the time lapsed before the request is closed. The system checks the threshold values before it uses the request to update data. When a threshold value is reached, the system closes the current request and the data transfer is continued using a new request.

You can only update data using a standard data transfer process if the request is closed. To schedule data update using a standard data transfer process in a process chain, use process type *Close Real-Time InfoPackage Request*. If you want requests to be closed by the process type, do not change the default threshold values in the InfoPackage.

6. Create a process chain that includes the processes listed below, activate the chain and schedule it:

   a. Start process: Specify the start conditions for the process chain.

      See: Start Process [Page 379]

   b. Close real-time InfoPackage request: Select the InfoPackage you have defined.

      See: Closing Requests Using Process Chains [Page 560]

   c. Data transfer process: Create the data transfer process using the DataSource you defined as the source and the InfoProvider you defined as the target.

      See: Creating Data Transfer Processes [Page 322]

   Include additional processes in your process chain, as required.

   For more information about process chain maintenance, see Creating Process Chains [Page 402].

Result

When the Web service returns data to the BI system, it is updated into the PSA table in an open request.

The scheduled process chain waits for the start event. The start event triggers the event to close the PSA request. When the Web service sends data to BI, the system checks whether the start event that closes the open request has been triggered. If this is the case, the open request is closed and the data transfer is continued using a new request. The closed request is updated to the InfoProvider using the data transfer process. Data is available for further update and processing or for reporting and analysis purposes.

4.3.1.3.4 Closing Requests Using Process Chains

Use

When you transfer data using a Web service or real-time data acquisition (using a SAPI and a Web service), the InfoPackage requests (also called PSA requests) remain open across several load processes. The requests are closed when the threshold values set in the InfoPackage are reached. The system opens new requests and data transfer is continued

using the new requests. With process type *Close Real-Time InfoPackage Request*, you can close an open PSA request before the threshold value is reached.

This means that you can use a Web service DataSource to send data to the PSA in BI and then use a standard data transfer process to update it further.

You can close requests in this way to perform regular analyses at set times on an InfoProvider that is down stream of a DataStore object that you are using for real-time data acquisition.

Procedure

1. In the process chain, choose process type *Close Real-Time InfoPackage Request*.
2. On the next screen, enter a technical name for the process variant and choose *Create*.
3. On the next screen, enter a description for the process variant and choose *Continue*.

   The maintenance screen for the process variant appears.
4. In the table, select the InfoPackage for which you want to close a request.
5. Choose *Save* and go back.

Do not schedule this process to take place more frequently than once an hour, otherwise performance is affected. If you schedule the process to take place more frequently, the system may generate so many requests that performance is affected.

Result

When the process chain is run, the system closes the PSA request, the DTP request and the change log requests when the start event is reached. The process chain does not wait until data is loaded; it closes any empty requests and ends the process with status *green*.

4.3.1.3.5 SOAP-Based Transfer of Data (3.x)

SOAP-based data transfer is still supported in data models with 3.x objects. Real-time data acquisition, however, is not possible for these models. For more information about the migration of existing data models and its objects, see Release and Upgrade Management [External].

Purpose

Data is generally transferred into SAP BW by means of a data request, which is sent from SAP BW to the source system (pull from the scheduler). You can also send the data to SAP BW from outside the system. This is a data push into SAP BW.

A data push is possible for various scenarios:

- Transferring Data Using the SOAP Service SAP Web AS [Page 567]
- Transferring Data Using Web Services [Page 571]
- Transferring Data Using SAP XI [Page 573]

In all three scenarios, data transfer takes place using transfer mechanisms that are sufficient for Simple Object Access Protocol (SOAP [External]); the data transfer is also XML-based.

The SOAP-based transfer of data is only possible for flat structures. You cannot transfer hierarchy data.

Process Flow

The data push is made to an inbound queue in SAP BW. SAP BW uses the delta queue of the service API as the inbound queue. To transfer the data, you generate a DataSource based on a file DataSource that has an interface for supplying the delta queue. The system generates an RFC-enabled function module for this XML DataSource. This updates the data to the delta queue for the XML DataSource. A prerequisite for updating to the delta queue is that you activate the data transfer to the delta queue beforehand.

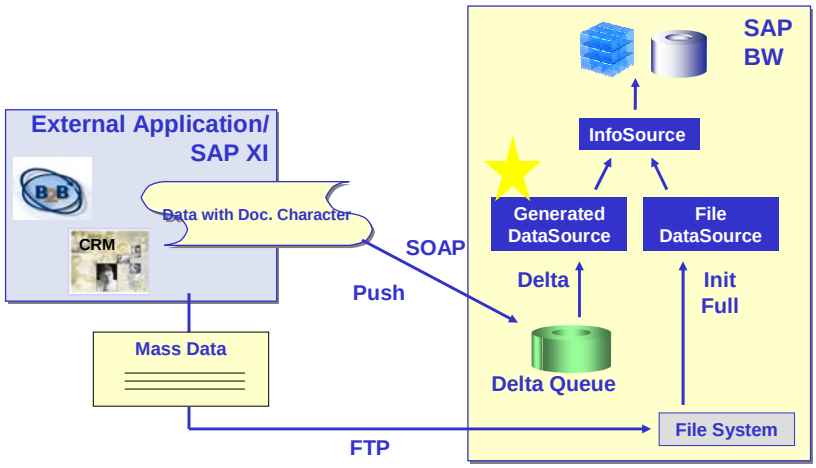
To make a data push into SAP BW using one of the three scenarios listed above possible, proceed as follows:

1. Create the XML DataSource.
   a. Create an InfoSource with flexible update and generate a file DataSource for it.
   b. Based on the file DataSource, generate an XML DataSource.
2. Activate the data transfer to the delta queue of the XML DataSource by initializing the delta process for the XML DataSource.

Result

You can use one of the three scenarios listed above to send the data to the delta queue in SAP BW. From there, you can process the data using the usual staging methods for deltas in SAP BW and then update it to the data targets.

The following figure outlines how data can be transferred to the SAP BW delta queue using a push in the delta process. For larger volumes of data, we recommend that you load the data using a full upload to the file DataSource. After the push, the data is checked for syntactic correctness, converted into ABAP fields, and then stored and collected in the delta queue of SAP BW. From there, the data is available for further processing in SAP BW.



4.3.1.3.5.1 XML DataSource (BW DataSource with SOAP Connection)

Definition

DataSource that is generated in SAP BW on the basis of a file-DataSource and that can be used to push data to SAP BW.

Use

With the help of the generated XML DataSource, you can transfer XML data into the SAP BW delta queue in order to continue to process it and to post it into the data targets you want.

Integration

The starting point for the generation of the XML DataSource is a file DataSource. It is used to characterize the data that is to be sent to the SAP BW.

You create the file DataSource using the definition of an InfoSource with flexible updating for a file source system. When the transfer rules are activated, you generate the file DataSource with the transfer structure. Now you can generate an XML DataSource in the maintenance of the transfer rules of the file DataSource using *Extras → Create BW DataSource with SOAP Connection.* This has the following properties:

- It is generated in a new namespace (`<xml-datasource> = 6a<file-datasource>`.
- The BW system itself is its source system (myself-connection)
- It is only intended for the loading of delta records, since the inbound queue is the delta queue in BW.
- It has an interface for supplying the delta queue.

  In doing so, the SAPI interface for supplying the delta queue is encapsulated by a DataSource-specific, RFC-compatible function module, which is generated for this purpose for the DataSource. Based on the RFC capability, the function module can be addressed externally (for example through a Web service, the http request handler of the SOAP service or the XI proxy runtime).

  The function module has the following properties:

Property	Technical name
Function group naming convention	/BIO/QI<xml-datasource>
Function module naming convention	/BIO/QI<xml-datasource>_RFC
Import parameter	<xml-datasource>
Table parameter	data

- The XML DataSource extraction structure is generated to suit the file DataSource transfer structure.
- The selectability of fields and the delta process are based on the file DataSource.

  If you have established the update mode *Additives Delta* for the file DataSource, the XML DataSource uses the ABR delta process (after, before, reverse), otherwise the XML DataSource uses the AIM delta process (after image).

4.3.1.3.5.2 Creating an XML DataSource

Use

An rfc-capable function module is generated with the XML DataSource that is required to push the data into SAP BW. You can also activate the data transfer to the delta queue – which is the inbound queue for the data push into SAP BW - for the XML DataSource.

Prerequisites

You have connected a file system to SAP BW as the source system.

Procedure

You are in the *Modeling* InfoSource tree of the Administrator Workbench.

1. Choose *InfoSources* → *Your Application Component* → *Context Menu (secondary mouse click)* → *Create InfoSource*...
2. Create an InfoSource with flexible updating (see Flexibly Updating Data from a Flat File [Page 287]).

   Flexible updating is a prerequisite for being able to set up the delta process for the file DataSource and being able to write the data to the delta queue.
3. Assign the file system as the source system to the InfoSource after activating the communication structure.

   The system generates a file DataSource with the same technical name as the InfoSource and assigns it to the InfoSource. The system also generates a proposal for the transfer structure and the transfer rules.
4. Change the transfer structure or the transfer rules where necessary.
5. Activate the transfer rules.

   The transfer structure and the DataSource are then likewise activated.

   When the transfer rules are activated, the menu option *Extras* → *Create BW DataSource with SOAP Connection* becomes active. Once you have activated the file DataSource you can create the XML DataSource.

   If you want to use a file DataSource that already exists, is not active or is not in the SAP namespace, the XML DataSource cannot be generated. You cannot generate an XML DataSource if the file DataSource is not active or if the file DataSource is in the SAP namespace.
6. In the InfoSource menu, choose *Extras* → *Create BW DataSource with SOAP Connection*

   The system generates the XML DataSource in its own namespace (`<xml-datasource> = 6A<file-datasource>`). The extraction structure is suitably generated for the file DataSource transfer structure. The field selectability and the XML DataSource delta process are likewise based on the file DataSource. The new DataSource is replicated and is assigned in the Administrator Workbench source system tree to the myself BW System in the *Delta Queue* application component under *Business Information Warehouse*.

For the XML DataSource, the system generates the RFC-compatible function module - matching the extract structure - which runs the data update in the delta queue.

7. Assign the XML DataSource to the InfoSource.

   You get to the transfer rules maintenance screen. The system automatically makes a proposal for the transfer rules on the basis of the file DataSource.

8. Activate the transfer rules.

Result

The XML DataSource with the generated function module for transferring data to the SAP BW is available to you.

Now you can activate data transfer to the delta queue for the DataSource and from now on, you can send data to SAP BW that is then written to the delta queue.

4.3.1.3.5.3 Activating Data Transfer to the Delta Queue

Use

Before you can send data to SAP BW with a push, you have to activate data transfer to the delta queue of an XML DataSource in SAP BW. If you send data to SAP BW afterwards, this data will be written to the delta queue and will then be available to you for further processing and updating. You can load data to the data targets using delta InfoPackages.

Prerequisites

You have created the XML DataSource.

Procedure

To activate data transfer to the delta queue, create an InfoPackage for your XML DataSource in order to initialize without a data request.

1. In the Modeling InfoSource tree of the Administrator Workbench , choose *InfoSources → Your Application Component → Your InfoSource for Requesting XML Data → myself BW System → Create InfoPackage*
2. Enter a description for your InfoPackage in the following dialog box. Select the XML DataSource and confirm your entries.
3. Edit the tab pages for the InfoPackage. Choose *Delta Process Initialization* mode on the *Update* tab page and select *Initialization without Data Transfer*.
4. Schedule the InfoPackage.

See also:

Maintaining InfoPackages [Page 343]

Scheduling InfoPackages [Page 365]

Result

Data transfer to the delta queue is now activated. This means that the XML DataSource is available as an entry in the delta queue. From this point on, the data that you send to SAP BW will be updated to the delta queue. You can check in transaction RSA7 whether or not the DataSource is available as an entry in the delta queue.

See also:

Checking the Delta Queue [Page 538]

4.3.1.3.5.4 Further Processing Data from the Delta Queue

Use

If the data in the delta queue for a DataSource is available in SAP BW, you can process it further using the usual staging methods for deltas and then you can update it to data targets in SAP BW.

Prerequisites

You have sent the data to the delta queue of a DataSource in SAP BW using the SOAP service, a Web services or SAP XI.

Procedure

1. Create an InfoPackage in the *Modeling* InfoSource tree of the Administrator Workbench under *InfoSources* → *Your Application Component* → *Your InfoSource for Requesting Data* → *myself BW System*, or change the InfoPackage you used for initialization.
2. Edit the InfoPackage and, on the *Update* tag page, choose *Delta Update* as the update mode.

   The selection of the update mode for the file DataSource (which you ran on the *DataSource/Transfer Structure* tab page in the transfer rules maintenance) affects the delta process for the data.

   If you have chosen the update mode *Additive Delta* (ODS and InfoCube) at the file DataSource, the delta process is the ABR process for the data. This means that the delta is created from after, before, and reverse images that have to be delivered from the data source.

   If you selected the update modi *Full Upload* (ODS and InfoCube) or *New Status for Changed Records* (only ODS object), the delta process is the AIM process for the data. This means that the delta is created from after images that have to be delivered from the data source.
3. Schedule the data request.

   We recommend that you do not load the data more frequently than once an hour from the delta queue.

See also:

Maintaining InfoPackages [Page 343]

Scheduling InfoPackages [Page 365]

Result

The data is available in the data target for further consolidation or evaluation.

4.3.1.3.5.5 ransferring Data Using the SOAP Service SAP Web AS

Purpose

XML [External] (eXtensible Markup Language) is a text-based, meta markup language that enables the description, exchange, display, and manipulation of structured data so that it can be used for a multitude of applications. You can send data from external applications in XML format using the Internet transfer protocol http directly to the SOAP [External] Service (Simple Object Access Protocol) of the SAP Web application server, which then integrates the data into SAP BW. In SAP BW, the data is written to the delta queue. You can process the data further with the available staging methods and then update it to the required data targets.

The transfer of XML data into SAP BW is suitable for regularly supplying SAP BW with **limited amounts of data for each call**; for example, the transfer of document data. Use the file DataSource to supply BW with larger amounts of data that are not transferred into BW using the XML interface.

Process Flow

As the basis for the solution, SAP BW uses the SOAP service provided with the SAP Web application server. You use this service to transfer XML data that is appropriate for the SOAP protocol to RFC-enabled function modules in the ABAP environment. Because it is RFC-enabled, the function module can be addressed automatically using one of the assigned HTTP handlers provided by SAP to support the SOAP log. The SOAP service checks the XML data for syntactical correctness and converts it into ABAP fields. The XML data has to be assigned according to an XML schema definition which is derived from the definition of the file or XML DataSource. The transfer of data into BW is performed by means of a push into the delta queue of the generated DataSource.

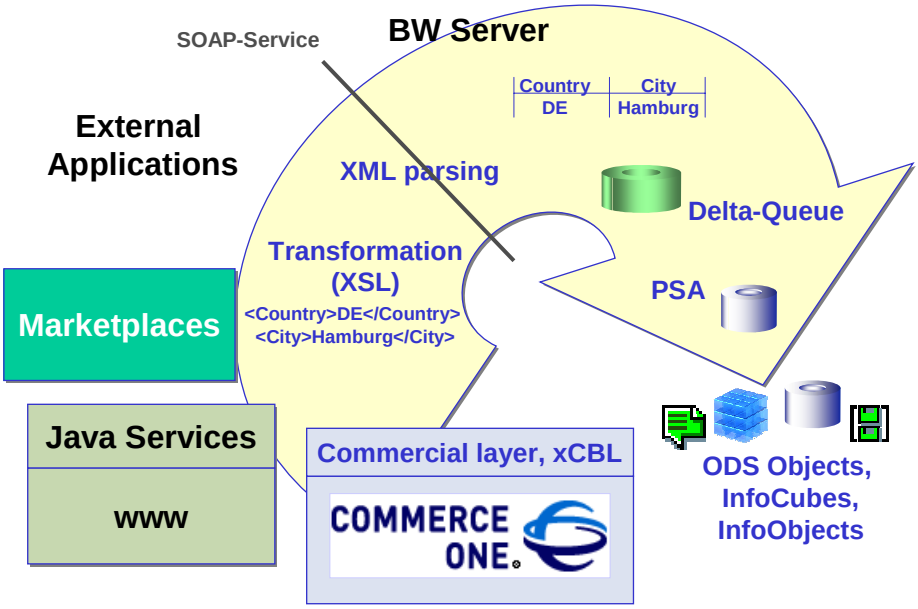
To enable data to be pushed using the SOAP service, perform the following steps in SAP BW.

1. Create a DataSource based on a file DataSource. When you generate the DataSource, an RFC-enabled function module is generated for data transfer. For more information, see XML DataSource [Page 563] and Creating XML DataSources [Page 564].
2. Activate the transfer of data to the SAP BW delta queue by initializing the delta process. For more information, see Activating Data Transfer to the Delta Queue [Page 565].

Result

You can send data to the SOAP service in XML format. From there you can collect data using the usual staging methods for deltas in SAP BW and then update it to the data targets. For more information, see Sending data to the SOAP Service [Page 568] and Further Processing Data from the Delta Queue [Page 566].

Example



4.3.1.3.5.5.1 Sending Data to the SOAP Service

Prerequisites

You have created an XML DataSource.

The XML data is available according to the XML schema of the file or XML DataSource of SAP BW.

You have activated the data transfer to the delta queue of the XML DataSource.

Procedure

Send the data to be loaded in XML format using the HTTP port provided under the name `/sap/bc/soap/rfc` to the SAP Web Application Server SOAP Service.

You find the relevant HTTP port in the Web services maintenance in your BW system. There you can check whether the SOAP service is active. For this, you need to choose *Go to → ICM Monitor* in the services maintenance (transaction **SICF**). There, you choose *Go to → Services.* The port to be used and the status of the service are displayed in the table for the HTTP log. If the service is deactivated, activate it by using *Go to → Service → Activate.*

You can find more information about SAP's Web services under Internet Communication Framework [External] and SOAP Runtime for SAP Web AS [External] in the connectivity documentation.

In the SOAP Service, the data is checked syntactically and reclassified in ABAP fields.

In the *XML* rubric at the Internet address **ifr.sap.com** in the document *Serialization for ABAP Data in XML* you can also find information about reclassifying XML data in ABAP fields.

The BW server SOAP interface can ensure **guaranteed delivery**, since an XML message is returned to the client whether successful or not. If the client has an error or no message (for example, update termination when sending a success message), the client can send the data again.

However, it can then ensure the **no guaranteed delivery only once** function, since there is no reconciliation on a transaction-ID level. On this basis you can determine that a data package was sent again “in error” and cannot be posted again. The update to an ODS object can, if the deltas with after-images (delta process AIM) are created, consistently deal with the data that is sent too often, as long as the serialization is secured. The serialization is the task of the client.

Afterwards, store the data in the delta queue.

Result

You can collect the data in the delta queue and can process it further using the usual staging methods for deltas in SAP BW and then post it to the data targets.

4.3.1.3.5.5.2 tructure of a SOAP Message

In the following example, you see the SOAP-compliant body of an HTTP post request for transferring data for the XML DataSource 6ATEST with the generated function module /BIC/QI6ATEST_RFC.

Correct structure of a SOAP message for transferring data to BW:

Components of the SOAP message	Explanations
`<?xml version="1.0" ?>`	Marks the header
`<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">`	Marks the beginning of the SOAP turnover
`<SOAP:Body>`	Marks the beginning of the body of data

`<rfc:_-BIO_-QI6ATEST_RFC xmlns:rfc="urn:sap-com:document:sap:rfc:functions">`	Calls the RFC-capable function module /BIO/QI<Techn. Name of XML DataSource> The / character must be replaced in the XML document by the character string _- so that it can be properly reclassified. Therefore, there is the function module name in the XML document `-BIO_-QI6ATEST_RFC`
`<DATASOURCE>6ATEST</DATASOURCE>`	Contains the technical name for the XML DataSource

`<DATA>`
`<item>`
`<VENDOR>JOHN</VENDOR>`
`<MATERIAL>DETERGENT</MATERIAL>`
`<DATE>20010815</DATE>`
`<UNIT>KG</UNIT>`
`<AMOUNT>1.25</AMOUNT>`
`</item>`
`<item>`
`<VENDOR>DAVID</VENDOR>`
`<MATERIAL>DETERGENT</MATERIAL>`
`<DATE>20010816</DATE>`
`<UNIT>ML</UNIT>`
`<AMOUNT>125</AMOUNT>`
`</item>`
`</DATA>`
	Contains the data in an XML wrapper
	The data package is opened with `<DATA>` and ended with `</DATA>`
	.
	
	The file only contains a data package in which different rows are included.
	The rows are opened with `<item>` and closed with `</item>`
	
	The field names must correspond to the XML DataSource technical names.

`</rfc:_-BIO_-QI6ATEST_RFC>`	Ends the RFC-capable function module /BIO/QI<Techn. Name of XML DataSource>.
`</SOAP:Body>`	Marks the end of the body of data
`</SOAP:Envelope>`	Marks the end of the SOAP envelope

4.3.1.3.5.6 Data Transfer Using Web Services

Purpose

You can generate Web services to load data based on function modules for XML DataSources. In this way, you can send data to the delta queue of SAP BW using the Web service. The Web services provide you with WSDL descriptions that can be used technologically independently of the push of data to SAP BW.

The BW server SOAP interface can ensure **guaranteed delivery**, since an XML message is returned to the client whether successful or not. If the client has an error or no message (for example, update termination when sending a success message), the client can send the data again.

However, it can then ensure the **no guaranteed delivery only once** function, since there is no reconciliation on a transaction-ID level. On this basis you can determine that a data package was sent again “in error” and cannot be posted again. The update to an ODS object can, if the deltas with after-images (delta process AIM) are created, consistently deal with the data that is sent too often, as long as the serialization is secured. The serialization is the task of the client.

Prerequisites

You are familiar with the Web service standards and technology.

Process

1. Create a DataSource based on a file-data source. When you generate the DataSource, an RFC-capable function module is generated for data transfer. You can find more information under XML DataSource [Page 563] and Creating XML DataSources [Page 564].
2. Activate the data transfer to the delta queue of SAP BW by initializing the delta process. You can find more information under Activating Data Transfer to the Delta Queue [Page 565].
3. You create an (ABAP) Web service for the previously generated function module and release it for SOAP runtime. You can find more information under Creating Web Services for Loading Data [Page 572]

Result

You can now use the Web service to send data to the delta queue of SAP BW. From there, you can collect the data using the usual staging methods for deltas in SAP BW and then post it to the data targets.

A WSDL description of the Web service, along with a test function to call the Web service is available in the Administration for SOAP Runtime [External] (transaction WSADMIN).

4.3.1.3.5.6.1 Creating a Web Service for Loading Data

Use

You can generate a Web service for the function module of the XML DataSource that provides a WSDL description that you can use independent of the communication technology to send data to SAP BW.

Prerequisites

You have created an XML DataSource.

You have activated the data transfer to the delta queue of SAP BW.

Procedure

1. Using the function library (transaction SE37), call the Web service creation wizard.

   To do this, select the desired function module in the function library and choose *Utilities* → *Generate Web Service* → *From the Function Module.*

2. Go through the following steps, shown in the wizard:

   a. Create a virtual interface.

      The virtual interface represents the interface between the Web Service and the outside.

   b. Choose the end point.

      The name of the function module that is to be offered as Web service is already entered here.

   c. Create the Web service definition.

      The Web service definition helps with assigning the Web service features, such as how security can be guaranteed in data transfer.

   d. Release the Web service.

      The wizard generates the object virtual interface and Web service definition in the object navigator.

      The function group that was generated when the XML DataSource was created is not transportable and is thus assigned to a local package. To prevent errors due to transports, make sure that the objects that were generated in the Web service creation wizard are assigned to a local non-transportable package.

      The Web service is released for the SOAP runtime.

3. In the virtual interface for the import parameter DATASOURCE, define the name of the XML DataSource as the fixed value.

   A separate function group is generated for each XML DataSource. It makes sense to pre-assign the parameter DATASOURCE with the name of the XML DataSource in the virtual interface of the Web service for which the function group was generated.

If you do not pre-assign the parameter, it will be necessary to transfer the data sent with the appropriate filled DataSource element, for example, by setting the value in the application that implements the Web service.

a. In the object navigator, choose the name of the package in which the Web service was created and choose *Enterprise Services* → *Web Service Library* → *Virtual Interfaces.*

b. Choose *Change* in the context menu for the virtual interface.

c. For the virtual interface, remove the flags *exposed* and *initial* and enter the name of the XML DataSource in apostrophes, for example **'6ADATASOURCENAME'**.

d. Activate the virtual interface.

Result

You have created a Web service for the XML DataSource and have release it for the SOAP runtime. You can now send data to the delta queue of SAP BW using the Web service.

Using *Web Service* → *WSDL* ( ) in the Administration for the SOAP Runtime [External] (transaction WSADMIN), you can call the WSDL description of the Web service. A test function to call and test the Web service is available under *Web Service* → *Web Service Homepage* ( ) (see Web Service Homepage [External]).

See also:

Creating ABAP Web Services [External]

Web Service Creation Wizard [External]

4.3.1.3.5.7 Data Transfer Using SAP XI

Purpose

You can realize cross-system business processes using the SAP Exchange Infrastructure (SAP XI). Within the overall architecture of SAP NetWeaver, SAP XI performs the tasks of process integration.

The integration of SAP XI and SAP BW allows you to use SAP XI to send data from various sources to the delta queue of SAP BW.

The integration of SAP XI and SAP BW offers the following advantages:

- Central maintenance of message flow between logical systems of your system landscape.
- Options for transformation of message content between sender and recipient

  Mappings help you to adapt values and structures of your message to the recipient. In this way, you can transfer different types of files to a SAP BW system using interface mapping. However, in any case, it is necessary to transform the data into a format that corresponds to the interface of the function module that is generated in SAP BW and used for data transfer. The function module contains a table parameter with a flat structure. This means that the data have to be transformed so that they fit to a flat structure in SAP BW.
- Using proxy communication with SAP BW

Proxies are executable interfaces generated in the application systems for communication with the SAP XI Integration Server. We recommend the use of proxies for communication with SAP BW because they guarantee **Full Quality of Service (Exactly Once in Order)**. They also guarantee that the data is delivered only once and in the correct sequence. The SAP XI Integration Server keeps the serialization as it was established by the sender.

Prerequisites

You are familiar with the concept, architecture and functions of SAP XI. You can find more information under SAP Exchange Infrastructure [External] in the NetWeaver documentation.

You have integrated SAP BW and SAP XI. You can find more information on this in the configuration guide of SAP XI on the SAP Service Marketplace at the Internet address **`service.sap.com/instguides`**.

Process

1. Create a XML DataSource in SAP BW based on a file-data source. When you generate the DataSource, an RFC-capable function module is generated for data transfer. You can find more information under XML DataSource [Page 563] and Creating XML DataSources [Page 564].
2. Activate the data transfer to the delta queue of SAP BW by initializing the delta process. You can find more information under Activating Data Transfer to the Delta Queue [Page 565].
3. You create an inbound and an outbound message interface in the Integration Repository of SAP XI.

   You can find more information under Design of Interfaces and Proxy Generation [External] in the documentation for SAP XI.

   If there is already an interface for data exchange in a system, you can import the interface description into the Integration Repository. You can find more information under Connection with Adapters and Imported Interfaces [External] in the documentation for SAP XI.

   The interface description in SAP BW is available in the form of the RFC-capable function module for the inbound message interface that was generated for your DataSource. To create the inbound message interface, you can import the function module into the SAP XI Integration Repository. You can find additional information under Import of Idocs and RFCs [External].

   - If you are using an existing SAP XI scenario, the outbound message interface is already in the Integration Repository. Then you only need to create the inbound message interface.
   - If you want to implement a new scenario, create an outbound message interface in addition to the inbound message interface.
4. You implement proxy generation for your inbound message interface in SAP BW.

   An ABAP object interface (inbound or server proxy) is generated in SAP BW for the inbound message interface.

   You can find more information under ABAP Proxy Generation [External] in the documentation for SAP XI.

We recommend proxy communication with SAP BW because it guarantees **Full Quality of Service (Exactly Once in Order)**.

5. You implement the generated ABAP object interface using an ABAP object class in SAP BW for recipient processing.

   You can find more information under ABAP Proxy Objects [External] in the documentation for SAP XI.

   The proxy runtime calls this processing automatically after receiving the appropriate message.

   The document How to…Integrate BW to XI describes such an implementation. You can find the document on the SAP Service Marketplace at the Internet address **`service.sap.com/bw`** → *Services & Implementation* → *HOW TO... Guides* → *Guide List SAP BW 3.x.*

6. If you have newly created the outbound message interface, you implement the data transfer according to your application case.

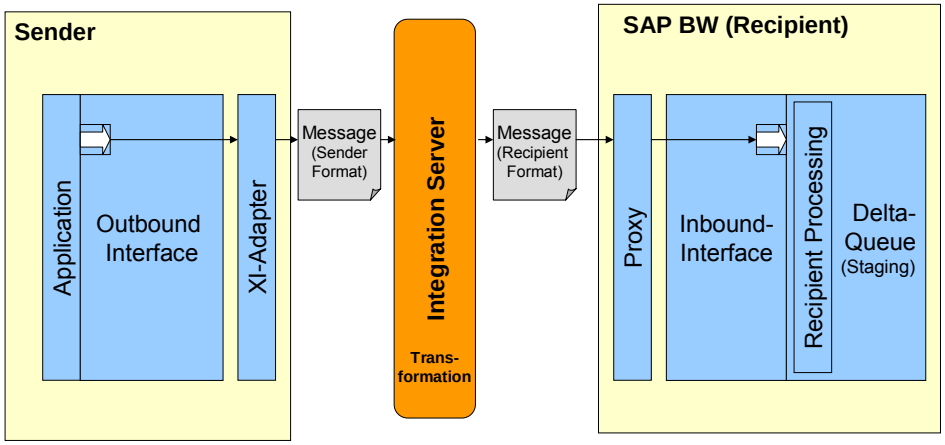
7. You implement the configurations in the Integration Directory of SAP XI that are relevant for message exchange. At the time of configuration, you set up the cross-system process for a concrete system landscape. The relevant objects are structure, organized and stored in the Integration Directory in the form of configuration objects.

You can find more information about the steps that you perform in SAP XI under Configuration [External] and Design [External] in the SAP XI documentation.

Result

You can now send data to the Integration Server of SAP XI, which transfers this data to SAP BW at runtime using proxy communication (see Proxy Runtime [External]). In SAP BW, the data is written to the delta queue. From there, you can collect the data using the usual staging methods for deltas in SAP BW and then post it to the data targets.

The following graphic illustrates how the interface-based processing of messages works:



Legend:

Parameters transferred to the interface

4.3.1.3.5.7.1 Example of Transferring XML Data Using SAP XI

If you want to load XML data into SAP BW using SAP XI, create an XML DataSource BW in SAP BW and activate data transfer to the delta queue. To generate the inbound message interface, import the function module of the XML Data Source generated for the DataSource into the SAP XI Integration Repository. For this inbound message interface, you generate a server proxy in SAP BW and implement the interface for recipient processing. Also create the outbound message interface in the SAP XI Integration Repository. After configuring SAP XI, you can send the XML data into a previously activated delta queue in SAP BW using the runtime infrastructure of SAP XI and you can process it further from there in SAP BW.

You can find a detailed description of this example of transferring XML data using SAP XI in the SAP Service Marketplace at the Internet address **`service.sap.com/BW`** → *Services & Implementation* → *HOW TO… Guides* → *Guide List SAP BW 3.x* → *How to… Integrate BW to XI*.

4.3.1.4 Transferring Data with UD Connect

Purpose

UD Connect (Universal Data Connect) uses the Application Server J2EE connectivity to enable the reporting and analysis of both SAP and non-SAP data. Using UD Connect, you can access just about all relational and multi-dimensional data sources. UD Connect transfers the data as flat data. Multi-dimensional data is converted to a flat format when UD Connect is used.

For the connection to data sources, UD Connect can use the JCA-capable (J2EE Connector Architecture) BI Java Connectors that are available for various drivers, protocols and providers as resource adapters.

- BI JDBC Connector
- BI ODBO Connector
- BI SAP Query Connector
- BI XMLA Connector

For more information, see BI Java Connectors [Page 589].

Prerequisites

You have installed the J2EE Engine with BI Java components. For more information, see the SAP NetWeaver Installation Guide on the SAP Service Marketplace at *service.sap.com/instguides.*

Terminology

UD Connect Source
UD Connect Sources are instances that can be addressed as data sources via the BI Java Connectors. Data sources that are included are relational data sources, OLAP data sources (multi-dimensional), XMLA-capable OLAP data sources (multi-dimensional) and SAP's own applications.

UD Connect Source Object
UD Connect source objects are multi-dimensional or relational data stores in the UD Connect sources, such as tables or cubes.

Source Object Element
Source object elements are the components of UD Connect source objects, such as fields for tables, characteristics and key figures of cubes.

Process Flow

1. Establish the connection to the data source with your relational or multi-dimensional source objects on the J2EE Engine, such as a relational database management system with tables and views.
2. Create RFC destinations on the J2EE Engine and in BI to enable communication between the J2EE Engine and BI. For more information, see the Implementation Guide for *SAP NetWeaver → Business Intelligence → UDI Settings by Purpose → UD Connect Settings*.
3. On the basis of the source object elements, model the required InfoObjects in BI.
4. In BI, determine a DataSource.

Result

You can now integrate the data for the source object into BI. You can either extract the data, load it into BI and physically store it there, or, as long as the prerequisites for this are fulfilled, you can read the data directly in the source using a VirtualProvider.

4.3.1.4.1 Creating a UD Connect Source System

Prerequisites

On the J2EE Engine of a SAP system, you have set the connection to the data source with your relational or multidimensional source objects.

You have created the RFC destinations on the J2EE Engine (of a SAP system) and in BI in order to enable communication between the J2EE Engine and BI. For more information, see the Implementation Guide for *SAP NetWeaver* → *Business Intelligence* → *UDI Settings by User Scenarios* → *UD Connect Settings*.

Procedure

1. In the source system tree in the Data Warehousing Workbench, choose *Create* in the context menu of the *UD Connect* folder.
2. Select the required *RFC Destination* for the J2EE Engine.
3. Specify a logical system name.
4. Select the type of connector.
5. Select the name of the connector.
6. Specify the name of the source system if it was not determined from the logical system name.
7. Choose *Continue*.

Result

When you use the created destinations, the settings that are required for communication between BI and the J2EE are created in BI.

4.3.1.4.2 Creating DataSources for UD Connect

Use

To transfer data from UD Connect sources into BI, metadata (the information about the source object and source object elements) has to be available in BI in the form of a DataSource.

Prerequisites

You have connected a UD Connect source system.

Note the following background information:

Using InfoObjects with UD Connect [Page 583]

Data Types and Their Conversion [Page 583]

Using SAP Namespace for Generated Objects [Page 584]

Procedure

You are in the DataSource tree in the Data Warehousing Workbench.

1. Select the application components in which the DataSource is to be created and choose *Create DataSource*.
2. In the next screen, enter a technical name for the DataSource, select the type of the DataSource and choose *Copy*.

   The DataSource maintenance screen appears.
3. Go to the *General* tab page.
   a. Enter descriptions for the DataSource (short, medium, long).
   b. If necessary, specify if the DataSource builds an initial non-cumulative and supplies potentially duplicate data records within a request.
4. Go to the *Extraction* tab page.
   a. Define the delta method for the DataSource.
   b. Specify whether you want the DataSource to support direct access to data.
   c. UD Connect does not support real-time data acquisition.
   d. The system displays *Universal Data Connect (Binary Transfer)* as the adapter for the DataSource.

      Choose *Properties* if you want to display the general adapter properties.
   e. Select the UD Connect source object.

      The connection to the UD Connect source is established. All source objects that are available in the selected UD Connect source can be selected using input help. Source objects can be multi-dimensional storage or relational objects.
5. Go to the *Proposal* tab page.

   The system displays the elements of the source object (for JDBC for example fields, or for XMLA and ODBO characteristics and key figures) and creates a mapping proposal for DataSource fields. The mapping proposal is based on the similarity of the names of the source object element and DataSource field and the compatibility of the respective data types.

   Note that source object elements can have a maximum of 90 characters. Both upper and lower case are supported.
   a. Check the mapping and change the proposed mapping as required. Assign the non-assigned source object elements to free DataSource fields.

      You cannot map elements to fields if the types are incompatible. The system produces an error message in this case.
   b. Under *Copy to Field List*, select the fields that you want to transfer to the field list for the DataSource. All fields are selected by default.
6. Maintain the *Fields* tab page.

   Here you edit the fields that you transferred to the field list of the DataSource from the *Proposal* tab page.
   a. Under *Transfer*, specify the decision-relevant DataSource fields that should be available for extraction and transferred to BI.
   b. If required, change the values for the key fields of the source.

      These fields are generated as a secondary index in the PSA. This is important in ensuring good performance for data transfer process selections, in particular with semantic grouping.
   c. If required, change the data type for a field.

d. Specify whether the source provides the data in the internal or external format.

e. If you choose an *External Format*, ensure that the output length of the field (external length) is correct. Change the entries, as required.

f. If required, specify a conversion routine that converts data from an external format into an internal format.

g. Select the fields for which you want to be able to set selection criteria when you schedule a data request using an InfoPackage. Data for this type of field is transferred in accordance with the selection criteria specified in the InfoPackage.

h. Choose the selection options (such as EQ, BT) that you want to be available for selection in the InfoPackage.

i. Under *Field Type*, specify whether the data to be selected is language-dependent or time-dependent, as required.

If you did not transfer the field list from a proposal, you can define the fields of the DataSource directly. Choose *Insert Row* and enter a field name. You can specify InfoObjects in order to define the DataSource fields. Under *Template InfoObject*, specify InfoObjects for the fields of the DataSource. This allows you to transfer the technical properties of the InfoObjects into the DataSource field.

Entering InfoObjects here does not equate to assigning them to DataSource fields. Assignments are made in the transformation. When you define the transformation, the system proposes the InfoObjects you entered here as InfoObjects that you might want to assign to a field.

7. Check, save and activate the DataSource.

8. Go to the *Preview* tab page.

   If you select *Read Preview Data*, the number of data records you specified in your field selection is displayed in a preview.

   This function allows you to check whether the data formats and data are correct.

Result

The DataSource is created and added to the DataSource overview for the UD Connect source system in the application component in the Data Warehousing Workbench. When you activate the DataSource, the system generates a PSA table and a transfer program.

You can now create an InfoPackage. You define the selections for the data request in the InfoPackage. The data can be loaded into the entry layer of the BI system, the PSA. Alternatively you can access the data directly if the DataSource allows direct access and you have a VirtualProvider in the definition of the data flow.

4.3.1.4.3 Creating DataSource 3.x

Use

Before you can transfer data from UD Connect sources into BI, you have to generate the metadata (the information about the source object and source object elements) in BI, in the form of a DataSource with a function module for extraction.

If your dataflow is modeled with objects based on the old concept (InfoSource 3.x, transfer rules 3.x, update rules 3.x), you can generate a DataSource 3.x for transferring data into BI from a database source system.

Prerequisites

You have modeled the InfoObjects that you want to use in the InfoSource and the data target or InfoProvider according to the UD Connect source object elements.

Note the following background information:

- Using InfoObjects with UD Connect [Page 583]
- Data Types and Their Conversion [Page 583]
- Using the SAP Namespace for Generated Objects [Page 584]

Procedure

You are in the *Modeling* InfoSource tree of the Administrator Workbench. First create in InfoSource and activate the communication structure. Then generate the generic DataSource using the wizard in the InfoSource maintenance.

1. Choose *InfoSources* → *Your Application Component* → *Context Menu (right mouse click)* → *Create InfoSource*.
2. Select the type of the InfoSource.
3. Under *InfoSource,* enter the technical name of the InfoSource, and assign a description and choose *Continue*.

   The system creates an InfoSource and shows it in the InfoSource tree under your application components.
4. In the context menu of the InfoSource, choose *Change* .

   The communication structure maintenance screen appears.
5. Using the previously modeled InfoObjects, create the communication structure (see Communication Structure [Page 276]).
6. Save and activate your communication structure
7. In the dialog box that follows you will be asked whether you wish to activate the dependent transfer program. Choose *No*.
8. In the InfoSource menu, choose *Extras* → *Create BW DataSource with UD Connect.*

   You get to the dialog box where you can assign a UD Connect source object to a DataSource and then generate the DataSource with the extractor. The fields for the DataSource are already being displayed in the table on the left side of the screen. The fields have the same name as the InfoObjects that you have used in the InfoSource.
9. Select the *RFC Destination* for the J2EE Engine.

   Make sure that the local server is running. If the local server is running and you cannot open the table for RFC destinations, start the local server again.

10. Choose the *UD Connect Source* in which the data that you wish to access is located.

    All available sources that are connected to the J2EE Engine are provided for selection in the input help. n instances are available per adapter.

11. Select the *UD Connect Source Object*.

    All available source objects that are available in the selected UD Connect source are provided for selection in the input help. Source objects can be multi-dimensional storage or relational objects.

    The system generates the name of the DataSource in the namespace `6B<Name of the source object><sequence number>`.

    In the *UD Connect Source Object* on the right side of the screen, the elements of the source object (for JDBC fields or for XMLA and ODBO characteristics and key figures) are displayed and the system generates a mapping proposal. The mapping proposal is based on the name similarity between source object element and DataSource field and the compatibility of the respective file types.

    Source object elements may be a maximum of 90 characteristics long. Upper and lowercase is supported.

    If you have entered the UD Connect source object manually, choose *Extract Source Object Elements* in order to generate the tables with the elements of the source object.

12. Check the mapping and change the proposed mapping as necessary. Assign the non-assigned source object elements to free DataSource fields.

    A mapping of elements to fields whose types are incompatible is not possible. The system reacts with an error message as needed.

13. Choose *Generate DataSource (for UD Connect).*

    - The system generates a DDIC structure for the generic DataSource and deletes any structures already in existence.
    - It creates the extraction function module and also deletes any currently existing modules.
    - In the BI myself system, the system generates a generic DataSource that uses the previously generated structure and the previously generated function module. The DataSource is created with the name `6B<Name of the source object><sequence number>.`
    - The DataSource is then replicated in BI.
    - The myself system is assigned to the InfoSource as source system as well as the DataSource.
    - The system generates a proposal for the transfer rules.

    Since the DDIC structure and the function module are located in the SAP namespace, the following entries can be requested during generation:

    - Developer and object key or
    - Developer key or
    - Object key and
    - Transport request

    If you do **not** make the requested entries, the generated infrastructure is not usable.

14. Change or complete the transfer rules as needed. For example, if a source object element is not assigned to a unit InfoObject, you can determine a constant for the unit, such as EUR for 0LOC_CURRCY (local currency).
15. Save and activate your transfer rules.

Recognizing Manual Entries

You can enter and change the *RFC Destination*, *UD Connect Source* and *UD Connect Source Object* manually. In order to validate these entries and the ones that depend on them, choose *Recognize Manual Entries.* For example, if you change the selected RFC destination, all of the field contents (*US Connect Source, UD Connect Source Object*, list of the source object elements) are invalid. If you choose *Recognize Manual Entries,* the dependent field contents are initialized and have to be maintained again.

Result

You have created the InfoSource and DataSource for data transfer with UD Connect. In the DataSource overview for the myself system, you can display the DataSource under the application components *Non-Assigned Nodes*.

4.3.1.4.3.1 Using InfoObjects with UD Connect

When modeling InfoObjects in BI, note that the InfoObjects have to correspond to the source object elements with regard to the type description and length description. For more information about data type compatibility, see Data Types and Their Conversion [Page 583].

The following restrictions apply when using InfoObjects:

- Alpha conversion is not supported
- The use of conversion routines is not supported
- Upper and lower case must be enabled

These InfoObject settings are checked when they are generated.

4.3.1.4.3.2 Data Types and Their Conversion

Based on the large number of possible UD Connect sources, the most diverse data types are possible in the output system. For this reason, a compatibility check is made at the time of generation of the UD Connect DataSource that is based on the type information supplied by the source systems. This attempts to decrease the probability of errors during the extraction process.

Following data type assignments are permitted:

Data Type in SAP BW	Data Type in the UD Connect Source
ACCP	C
CHAR	All except X and b
CUKY	C
CURR	P, I
DATS	D, g

DEC	P, I
FLTP	F, I
INT1	B
INT2	S
INT4	I
LCHR	G, V
NUMC	I
PREC	b
QUAN	I, P
SSTR	C
STRG	g
TIMS	T
VARC	All except X, P, F
UNIT	C, g

Abbreviations for the data types of UD Connect sources:

C – character

X – hexadecimal

P – packed decimal, decimal

I – integer

n – numeric string

D – date

b – tiny int

g, G – long string

F – float

s – small int

V – variable character

T – time

4.3.1.4.3.3 Using SAP Namespace for Generated Objects

The program-technical objects that are generated during generation of a DataSource for UD connect can be created in transportable or local format. Transportable means that the generated objects can be transferred to another SAP BW system using the correction and transport system. The transportability of an object depends on, among other things, in which namespace it is created.

The delivery status allows for the generation of transportable objects in the SAP namespace. If this appears to be too laborious (see the dependencies listed below), there is also the option of switching to generation of local objects. To do this, you run the **RSSDK_LOCALIZE_OBJECTS** report in the ABAP editor (transaction: SE 38). Then the system switches to local generation. The objects generated afterward are **not** transportable. If

the report is executed again, the generation is changed back to transportable. The status of already generated objects does **not** change. All new objects are created as transportable.

If you need to work with transportable objects, you should be aware of the following dependencies:

- System changeability
  - These objects can only be generated in systems whose system changeability permits this. In general, these are development systems, because productive systems block system changeability for security reasons.
  - If a classic SAP system landscape of this type exists, then the objects are created in the development system and assigned to package **RSSDK_EXT**. This package is especially designated for these objects. The objects are also added to a transport request that you create or that already exists. After the transport request is finished, it is used to transfer the infrastructure into the productive environment.
- Key

  Because the generated objects are ABAP development objects, the user must be authorized as a developer. A developer key must be procured and entered. Generation requires the customer-specific installation number and can be generated online. The system administrator knows this procedure and should be included in the procurement. The key has to be procured and entered exactly once per user and system.

  Because the generated objects were created in the SAP namespace, an object key is required. Like the developer key, this is customer specific and can also be procured online. The key is to be entered exactly once per object and system. Afterwards, the object is released for further changes as well. Further efforts are not required if there are repeated changes to the field list or similar.

4.3.1.4.4 Using Emulated 3.x DataSources

Use

You can display an emulated 3.x DataSource in DataSource maintenance in BI. Changes are not possible in this display. In addition, you can use emulation to create the (new) data flow for a 3.x DataSource with transformations, without having to migrate the existing data flow that is based on the 3.x DataSource.

We recommend that you use emulation before migrating the DataSource in order to model and test the functionality of the data flow with transformations, without changing or deleting the objects of the existing data flow. Note that use of the emulated Data Source in a data flow with transformations has an effect on the evaluation of the settings in the InfoPackage. We therefore recommend that you only use the emulation in a development or test system.

Constraints

An emulated 3.x DataSource does not support real-time data acquisition, using the data transfer process to access data directly, or loading data directly (without using the PSA).

Prerequisites

If you want to use transformations in the modeling of the data flow for the 3.x DataSource, the transfer rules and therefore the transfer structure must be activated for the 3.x DataSource. The PSA table to which the data is written is created when the transfer structure is activated.

Procedure

To display the emulated 3.x DataSource in DataSource maintenance, highlight the 3.x DataSource in the DataSource tree and choose *Display* from the context menu.

To create a data flow using transformations, highlight the 3.x DataSource in the DataSource tree and choose *Create Transformation* from the context menu. You also use the transformation to set the target of the data transferred from the PSA.

To permit a data transfer to the PSA and further updating of the data from the PSA to the InfoProvider, select the DataSource 3.x in the DataSource tree and choose *Create InfoPackage* or *Create Data Transfer Process* in the context menu. We recommend that you use the processes for data transfer to prepare for the migration of a data flow and not in the production system.

Result

If you defined and tested the data flow with transformations using the emulation, you can migrate the DataSource 3.x after a successful test.

4.3.1.4.5 Using Relational UD Connect Sources (JDBC)

Aggregated Reading and Quantity Restriction

In order to keep the data mass that is generated during UD Connect access to a JDBC data source as small as possible, each select statement generated by the JDBC adapter receives a *group by* clause that uses all recognized characteristics. The recognized key figures are aggregated. What is recognized as a key figure or characteristic and which methods are used for aggregation depends on the properties of the associated InfoObjects modeled in SAP BW for this access.

The amount of extracted data is not restricted. To prevent exceeding the storage limitations of the J2EE server, packages with around 6,000 records are transferred to the calling ABAP module.

Use of Multiple Database Objects as UD Connect Source Object

Currently only one database object (table, view) can be used for a UD Connect Source. The JDBC scenario does not support joins. However, if multiple objects are used in the form of a join, a database view should be created that provides this join and this object is to be used as a UD Connect source object.

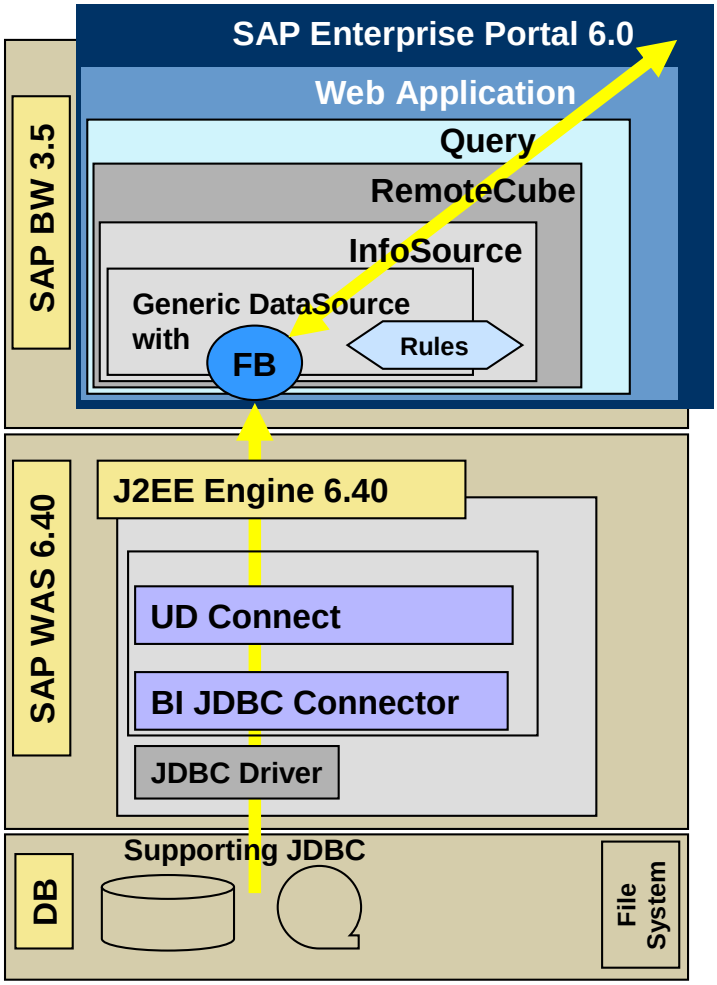
The view offers more benefits:

- The database user selected from SAP BW for access is only permitted to access these objects.
- Using the view, you can run type conversions that cannot be made by the adapter (generation of the ABAP data type DATS, TIMS etc.)

4.3.1.4.6 Dataflow 3.x Example: JDBC Source with Transaction Data

In the following example, we are assuming that the prerequisites for the use of an SAP RemoteCube [Page 166] are fulfilled. In this way, you can define a query with direct access to the transaction data in the UD Connect source. The data is not physically stored in BI.

A database management system (DBMS) with tables is used as the UD Connect source and views are used as the UD Connect source objects. In order to be able to use this source you have to install the appropriate JDBC driver for your DBMS provider on the J2EE Engine of the SAP Web AS. After that you can configure the BI JDBC Connector, that is, the connection between the J2EE Engine and the DBMS. In BI, you create an InfoSource with flexible update based on InfoObjects that are compatible with the view or table fields of the DBMS. For this InfoSource, you generate a generic DataSource for access to data in the DBMS. Select a table or a view for the DBMS and assign the fields to the DataSource fields. You use an SAP RemoteCube (that you have generated from the InfoSource) to define a query in BI. You can use the data in the table or view for immediate analysis; you do not have to load it into BI. You can use all the analysis tools of the Business Explorer (query in BEx Analyzer or Web application), or you can run an analysis in the portal.



Inversion of transfer rules with direct access via SAP RemoteCube

If you have established transfer rules, they will be inverted at the value selection in a query. When accessing data in the data source (in your case, table or view), the system reverses the order of the transfer rules.

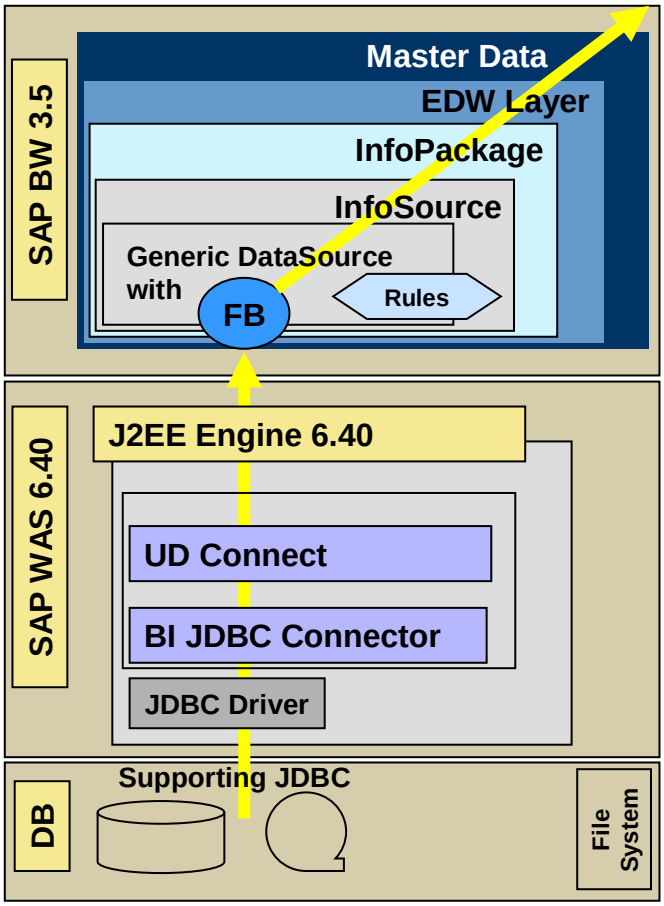
For example, if you select the period 1.2002 until 5.2002 (characteristic 0FISCPER), and the DataSource contains this information in two fields for year and period (the fields year and period are mapped with transfer rules on 0FISCPER), BI inverts the transfer rules and divides the selection into year 2002 and period 1,...,5. This selection is passed on to the table or the view and then sent on to BI. In BI, year and period are combined again into 0FISCPER in the transfer rules and the data is displayed according to the selection in the query.

4.3.1.4.7 Dataflow 3.x Example: JDBC Source with Master Data

In the following example, master data are extracted from the UD Connect source, loaded into BI and physically stored there.

A database management system (DBMS) with tables is used as the UD Connect source and views are used as the UD Connect source objects. In order to be able to use this source you have to install the appropriate JDBC driver for your DBMS provider on the J2EE Engine of the SAP Web AS. After that you can configure the BI JDBC Connector, that is, the connection between the J2EE Engine and the DBMS. In BI, you create an InfoSource on InfoObjects that are compatible with the view or table fields of the DBMS. For this InfoSource, you generate a generic DataSource for access to data in the DBMS. Select a table or a view for the DBMS and assign the fields to the DataSource fields. You create an InfoPackage for the InfoSource and use it to determine parameters for the data transfer into BI and to load data into BI. Data is stored physically in BI and can be used for analysis purposes in the portal or using the Business Explorer tools.

There is no inversion of the transfer rules in this case because when you make selections in the query, you are accessing the data that is physically stored in the BI Enterprise Data Warehouse Layer and has already been transformed and accessed.

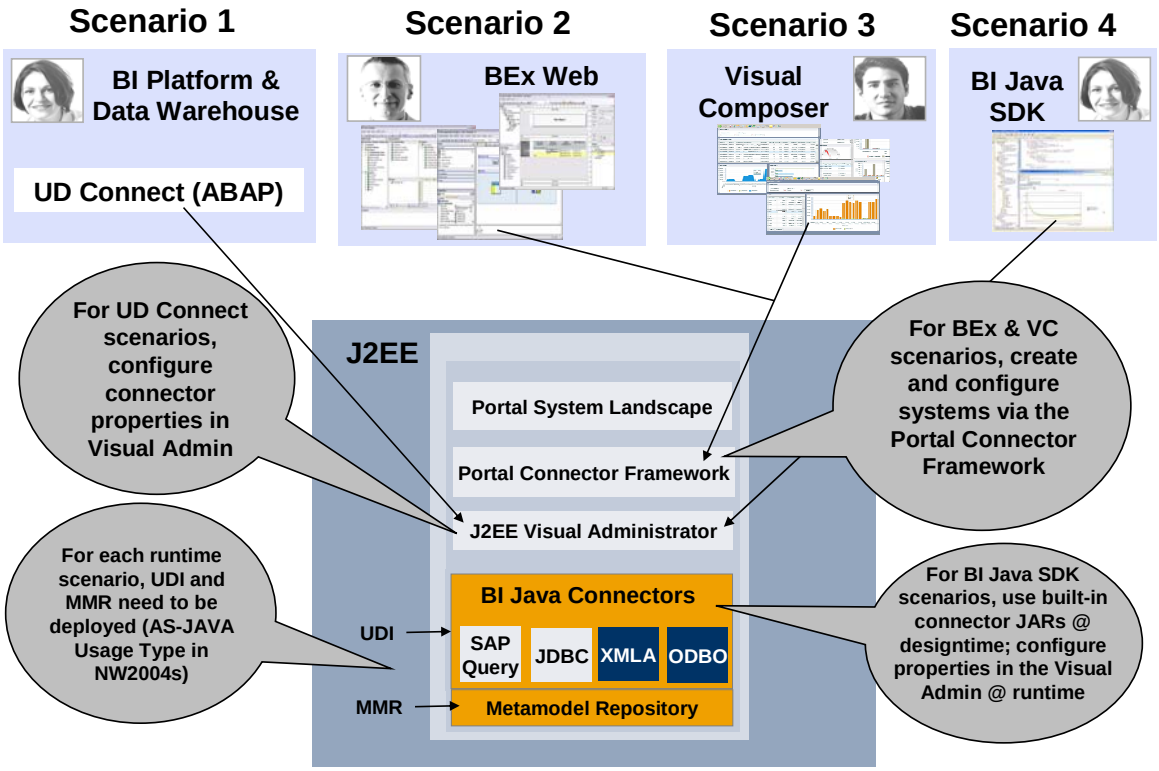


4.3.1.4.8 BI Java Connectors

Purpose

The BI Java Connectors are a group of four JCA (J2EE Connector Architecture)-compliant resource adapters that implement the BI Java SDK's APIs and allow you to connect the applications you build with the SDK to heterogeneous data sources. You can also use the BI Java Connectors to make these data sources available in SAP BI systems via UD Connect, or to create systems in the portal to use in BEx Web or Visual Composer scenarios.

The following diagram outlines the potential usage scenarios for the BI Java Connectors:



As illustrated, you can use the BI Java Connectors to create systems for use in four different scenarios. Since the BI Java Connectors are distributed as part of SAP Universal Data Integration (UDI), these are often referred to as UDI scenarios:

- **Scenario 1: UD Connect**

  On the BI platform, you can use UD Connect to make data from systems based on the BI Java Connectors available in SAP BI. For more information, see Transferring Data with UD Connect [Page 576].

  To configure the BI Java Connectors for this scenario, refer to the following path in the *SAP Customizing Implementation Guide*: *SAP NetWeaver → Business Intelligence → UDI Settings by Purpose → UD Connect Settings*, and follow the instructions in Configuring the BI Java Connectors [Page 592] to configure the connector properties.

- **Scenario 2: BEx Web**

  You can use data from systems based on the BI Java Connectors in BEx Web Applications. For more information, see BEx Web [External] and Access to External Data Sources [External].

  To configure the BI Java Connectors for this scenario, refer to the following path in the *SAP Customizing Implementation Guide*: *SAP NetWeaver → Business Intelligence → UDI Settings by Purpose → Integration in BEx Web*, and see Running the System Landscape Wizard [External] and Editing Systems [External] to configure the systems on the portal.

- **Scenario 3: Visual Composer**

You can use data from systems based on the BI Java Connectors in Visual Composer, the portal-based visual modeling application. For more information, see the SAP NetWeaver Visual Composer Modeler's Guide [External].

To configure the BI Java Connectors for this scenario, see the SAP NetWeaver Visual Composer Installation & Configuration Guide [External], and see Running the System Landscape Wizard [External] and Editing Systems [External] to configure the systems on the portal.

- **Scenario 4: BI Java SDK**

  You can build custom Java applications based on data in systems created with the BI Java Connectors. For more information, see BI Java SDK [External].

  To configure the BI Java Connectors for this scenario, see Configuring the BI Java Connectors [Page 592].

Features

Four BI Java Connectors are available:

- BI JDBC Connector [Page 595], for connecting to relational JDBC data sources
- BI ODBO Connector [Page 597], for connecting to OLE DB for OLAP-compliant data sources
- BI SAP Query Connector [Page 600], for connecting to data from SAP operational applications
- BI XMLA Connector [Page 601], for connecting to XMLA-compliant data providers such as SAP BI Systems

Connector Overview

Connector	Access To	Technology Based On	System Requirements
BI JDBC Connector	Relational data sources: over 170 JDBC drivers *Examples:* Teradata, Oracle, Microsoft SQL Server, Microsoft Access, DB2, Microsoft Excel, text files such as CSV	Sun's JDBC (Java Database Connectivity) -- the standard Java API for Relational Database Management Systems (RDBMS)	JDBC driver for your data source
BI ODBO Connector	OLAP data sources: OLE DB for OLAP-compliant data sources *Examples:* Microsoft Analysis Services, SAS, Microsoft PivotTable Services	Microsoft's OLE DB for OLAP -- the established industry-standard OLAP API for the Windows platform	• Microsoft Windows 2000 / NT / XP or greater • Microsoft Data Access Components (MDAC) 2.6 or greater • ODBO driver

			for your data source
BI SAP Query Connector	SAP operational applications *Examples:* Data in transactional systems such as Ad-Hoc, and Operational Reporting	SAP Query -- a NetWeaver component that allows you to create custom reports without any ABAP programming knowledge	SAP JCo
BI XMLA Connector	XMLA-compliant OLAP data sources *Examples:* MS Analysis Services, Hyperion, MicroStrategy, and SAP BW 3.x	Microsoft's XMLA (XML for Analysis) -- Web services-based, platform-independent access to OLAP providers. Exchanges analytical data between a client application and a data provider working over the Web, using a SOAP-based XML communication API.	none

See also:

- To configure a BI Java Connector on the server using the Visual Administrator, see Configuring the BI Java Connectors [Page 592].
- To create a system on the portal using a BI Java Connector, see Creating Systems [External].
- For more information about the J2EE Connector Architecture (JCA), see ***http://java.sun.com/j2ee/connector/***
- For information about the BI Java SDK and its connection architecture, refer to the ***index.html*** file inside of the SDK distribution package
- Read about the BI Java SDK [External]

4.3.1.4.8.1 Configuring the BI Java Connectors

Use

To prepare a data source for use with the BI Java SDK or with UD Connect, you must first configure the properties of the BI Java Connector used to connect to the data source. You do this in the SAP NetWeaver Application Server's Visual Administrator, following the steps below.

To create and configure systems in the portal for use in BEx Web and Visual Composer scenarios, see Running the System Landscape Wizard [External] and Editing Systems [External] in the NetWeaver Portal System Landscape documentation.

Prerequisites

- In order to configure the properties for a data source based on a BI Java Connector, the connector's resource adapter archive (RAR file) (delivered as part of Universal Data Integration, or UDI) and the Metamodel Repository (MMR) upon which the connectors rely must first be deployed to the server. UDI and MMR are part of usage type AS-Java (Application Server – Java) in NetWeaver 2004s.
- Check the documentation for the connector you wish to configure for any additional prerequisites (you'll also be referring to these documents for the list of specific properties to configure):
  - BI JDBC Connector [Page 595]
  - BI ODBO Connector [Page 597]
  - BI SAP Query Connector [Page 600]
  - BI XMLA Connector [Page 601]

Procedure

1. Start the Visual Administrator:

   UNIX: On your central instance host, change to the admin directory `/usr/sap/<SAPSID>/<instance_number>/j2ee/admin` and execute `go.sh`.

   Windows: On your central instance host, change to the admin directory `\usr\sap\<SAPSID>\<instance_number>\j2ee\admin` and execute `go.bat`.

2. On the *Cluster* tab, choose *Server x* → *Services* → *Connector Container*.
3. Locate your connector in the *Connector Container* tree and double-click it to open the connector definition:

   BI JDBC Connector: SDK_JDBC under the node sap.com/com.sap.ip.bi.sdk.dac.connector.jdbc

   BI ODBO Connector: SDK_ODBO under the node sap.com/com.sap.ip.bi.sdk.dac.connector.odbo

   BI SAP Query Connector: SDK_SAPQ under the node sap.com/com.sap.ip.bi.sdk.dac.connector.sapq

   BI XMLA Connector: SDK_XMLA under the node sap.com/com.sap.ip.bi.sdk.dac.connector.xmla.

4. On the *Runtime* tab (in the right frame), choose *Managed Connection Factory* → *Properties*.
5. Select and edit each property according to *Connector Properties* tables in the documents below:

   BI JDBC Connector [Page 595]

   BI ODBO Connector [Page 597]

   BI SAP Query Connector [Page 600]

   BI XMLA Connector [Page 601]

6. After configuring each property, choose *Add* to transfer the changes to the active properties list.
7. *Save* the settings.

For the BI JDBC Connector only:

Still in the *Connector Container* service, configure a reference to your data source's JDBC driver using the steps below:

Select the BI JDBC Connector in the *Connectors* tree.

Choose the *Resource Adapter* tab.

In the *Loader Reference* box, choose *Add* to add a reference to your JDBC driver.

Enter `library:<jdbc driver name>` and choose *OK*.

The `<jdbc driver name>` is the name you entered for your driver when you loaded it (see *Prerequisites* in BI JDBC Connector [Page 595]).

Save the settings.

For the BI XMLA Connector only:

If you are using the SAPLOGONTICKET authentication method, configure BI to accept

user/SAPLogonTicket pairs for authentication (instead of only user/password pairs) using the procedures below:

Configure SSO between the J2EE engine and the SAP BI system (so that ABAP accepts SSO from the J2EE engine) using the instructions in the following document:

Configuring SAP Web AS ABAP to Accept Logon Tickets from the J2EE Engine [External]

Configure SSO between the SAP BI system and the J2EE engine (so that the J2EE engine accepts SSO from ABAP) using the instructions in the following document:

Configuring the J2EE Engine to Accept Logon Tickets [External]

If an error occurs when configuring and testing the BI XMLA Connector, apply **SAP Note 863672**.

For more information on using the Connector Container service, see Connector Container Service [External].

Result

Your BI Java Connector properties are configured and your data source is ready to use.

Testing the Connections

After you configure a BI Java Connector, you can perform a rough installation check by displaying the page for a given connector in your server. Perform the tests for a given connector by visiting the URLs in the table below:

Connector Test Servlets

Connector	URL	Successful Result
BI JDBC Connector	http://<host>:<port>/TestJDBC_Web/TestJDBCPage.jsp	A list of tables is displayed

BI ODBO Connector	http://<host>:<port>/TOdbo/servlet/TestOdbo	A list of cubes is displayed
BI SAP Query Connector	http://<host>:<port>/TSapq/servlet/TestSapq	A list of tables is displayed
BI XMLA Connector	http://<host>:<port>/TXmla/servlet/TestXmla	A list of cubes is displayed

These tests are designed to work with the default installation of the BI Java Connectors. Cloned connectors with new JNDI names are not tested by these servlets.

JNDI Names

When engineering applications with the BI Java SDK, refer to a connector by its JNDI name:

Connector JNDI Names

Connector	JNDI Name
BI JDBC Connector	SDK_JDBC
BI ODBO Connector	SDK_ODBO
BI SAP Query Connector	SDK_SAPQ
BI XMLA Connector	SDK_XMLA

For code samples, refer to the *Connection Interfaces* Javadocs package in the documentation set delivered with the BI Java SDK.

Cloning the Connections

You can clone an existing connection by using the *Clone* button in the toolbar.

For Universal Data Connect (UD Connect) only:

When entering the resource adapter name during the cloning process, you must prefix the JNDI name with SDK_ and use only all capital letters in the name in order for UD Connect to properly recognize the connector.

4.3.1.4.8.2 I JDBC Connector

Use

Sun's JDBC (Java Database Connectivity) is the standard Java API for Relational Database Management Systems (RDBMS). The BI JDBC Connector allows you to connect applications built with the BI Java SDK to over 170 JDBC drivers, supporting data sources such as Teradata, Oracle, Microsoft SQL Server, Microsoft Access, DB2, Microsoft Excel, and text files such as CSV. This connector is fully compliant with the J2EE Connector Architecture (JCA).

You can also use the BI JDBC Connector to make these data sources available in SAP BI Systems via UD Connect, or you can create systems in the portal based on this connector.

The connector adds the following functionality to existing JDBC drivers:

- Uniform connection management that integrates with user management in the portal

- Uniform metadata service, by implementing JMI capabilities based on CWM
- Query model independent of SQL dialect of the underlying data source

The JDBC Connector implements the BI Java SDK's IBIRelational interface.

Prerequisites

The BI JDBC Connector supports any JDBC-compliant data source.

If you have not already done so, you must first deploy your data source's JDBC driver to the server:

1. Start the Visual Administrator.
2. On the *Cluster* tab, select *Server x* → *Services* → *JDBC Connector*.
3. In the right frame, select the *Drivers* node on the *Runtime* tab.
4. From the icon bar, choose *Create New Driver or Data source*.
5. In the *DB Driver* field in the *Add Driver* dialog box, enter a name for your JDBC driver.
6. Navigate to your JDBC driver's JAR file and select it.
7. To select additional JAR files, select *Yes* when prompted, and when finished, select *No*.

For more information, see JDBC Connector Service [External].

Connector Properties

Refer to the table below for the required and optional properties to configure for your connector:

BI JDBC Connector Properties

Property	Description	Examples
UserName	Data source username. User must have at least read access to the data source.	(your user name)
Password	Data source password.	(your password)
URL	URL string specifying the location of a database (used by the java.sql.DriverManager to determine which driver to use).	`jdbc:inetdae7:domain:port?database=mydatabase`
DriverName	Class name of the JDBC driver used for this connection.	`com.inet.tds.TdsDriver`
FixedCatalog	Restriction of metadata access to metadata contained in specified catalog. *Optional*.	`null` (no restriction) `xyz` (restrict access to catalog "xyz")
FixedSchema	Restriction of metadata access to metadata	`null` (no restriction)

	contained in specified schema. *Optional*.	`xyz` (restrict access to schema “xyz”)
Language	Two-letter abbreviation of language. Specifies the language of exceptions evoked on the BI Java SDK layer. JDBC databases themselves do not support this property. *Optional*.	`EN`=English `DE`=German

See also:

- To configure BI Java Connector properties in the SAP NetWeaver Application Server's Visual Administrator, see Configuring the BI Java Connectors [Page 592].
- To create a system on the portal using the BI JDBC Connector, see Creating Systems [External] and Editing BI JDBC System Properties [External].
- For information about the BI Java SDK and its connection architecture, refer to the ***index.html*** file inside of the SDK distribution package
- For more information about Sun's J2EE Connector Architecture, see ***http://java.sun.com/j2ee/connector/***
- For more information about Sun’s JDBC, see ***http://java.sun.com/products/jdbc/***

4.3.1.4.8.3 I ODBO Connector

Use

Microsoft's OLE DB for OLAP is the established industry-standard OLAP API for the Windows platform. The BI ODBO Connector is based on OLE DB for OLAP, abbreviating it as "ODBO." This connector allows you to connect applications built with the BI Java SDK to ODBO-compliant OLAP data sources such as Microsoft Analysis Services, SAS, and Microsoft PivotTable Services. This connector is fully compliant with the J2EE Connector Architecture (JCA).

You can also use the BI ODBO Connector to make these data sources available in SAP BI Systems via UD Connect, or you can create systems in the portal based on this connector.

The BI ODBO Connector uses Microsoft's ADO (ActiveX Data Objects) and ADO MD (ActiveX Data Objects Multidimensional) to support connectivity to OLAP data sources. ADO provides access to the schema object; ADO MD adds easy access to multidimensional data by extending ADO with objects specific to multidimensional data, such as the cubes and cellsets. With ADO and ADO MD, you can browse multidimensional schema, query a cube, and retrieve the results, thus providing convenient access to OLAP data from languages such as Microsoft Visual Basic, Microsoft Visual C++, and Microsoft Visual J++. Like ADO, ADO MD uses an underlying OLE DB provider to gain access to data.

The ODBO Connector implements the BI Java SDK's IBIOlap interface.

Prerequisites

- Windows 2000 / NT / XP (or greater) systems only

- Microsoft Data Access Components (MDAC) 2.6 or greater
- ODBO (OLE DB for OLAP) driver for your data source

Connector Properties

Refer to the table below for the required and optional properties to configure for your connector:

BI ODBO Connector Properties

Property	Description	Examples
UserName	Data source user name. User must have at least read access to the data source.	(your user name)
Password	Data source password.	(your password)

ConnString	Connection string information such as provider name, file name, remote provider, remote server, and URL.
	
	Note: For SAP BI OLAP providers, to completely suppress the SAP Logon screen at runtime, be sure to also provide values for the `UserName`, `Password`, and `Language` properties. If you want to evoke the logon screen, you may specify the `ConnString` property only.
		Local cube:
		`Provider=MSOLAP;Location="c:\public\SalesOverview.cub"`
		
		Microsoft Analysis Server:
		`Provider=MSOLAP;data source=palbiteam`
		
		BW OLAP provider:
		`Data Source=BWP;`
		`Provider=MDrmSAP;`
		`SFC_CLIENT=010;`
		`SFC_LANGUAGE=EN`

Language	Two-letter abbreviation of language. Sets the language for your data source (if supported), and also specifies the language of exceptions evoked on the BI Java SDK layer. Optional. Default is `EN`.	`EN`=English `DE`=German
LogonAsUserID (see note below table)	Username of a given windows domain account. *Optional*. Empty string	(your user name)

	means property not set.	
LogonAsUserPassword (see note below table)	Password for a given windows domain account. *Optional*. Empty string means property not set.	(your password)
LogonAsUserDomain (see note below table)	Windows domain account name. *Optional*. Empty string means property not set.	(a Windows domain name)

Using a specific Windows domain account:

You may wish to set a specific Windows domain account in certain situations where the BI ODBO Connector needs to use security credentials different from the credentials used by J2EE server itself. Examples of this include when the OLE DB driver for Microsoft Analysis Server supports only Windows domain security and ignores credentials supplied in the connection string, if any, or with a connection to MS SQL Server where MS SQL Server is configured for Windows authentication only.

Configuring these properties will result in the BI ODBO Connector attempting to impersonate the specified Windows domain account when establishing a connection to the target system. The J2EE server will create a local COM component using the account information provided, and this component, in turn, will access the remote computer (such as that which hosts the database server). These properties are ignored if `LogonAsUserID` and/or `LogonAsUserDomain` contain an empty string or are absent.

See also:

- To configure BI Java Connector properties in the SAP NetWeaver Application Server's Visual Administrator, see Configuring the BI Java Connectors [Page 592].
- To create a system on the portal using the BI ODBO Connector, see Creating Systems [External] and Editing BI ODBO System Properties [External].
- For information about the BI Java SDK and its connection architecture, refer to the ***index.html*** file inside of the SDK distribution package.
- For information about the SAP NetWeaver Business Intelligence OLE DB for OLAP interface, see OLE DB for OLAP [External].
- For more information about Sun's J2EE Connector Architecture, see ***http://java.sun.com/j2ee/connector/***
- For more information about Microsoft's ADO, see: ***http://msdn.microsoft.com/library/default.asp?URL=/library/psdk/dasdk/mdac3sc7.htm***

4.3.1.4.8.4 I SAP Query Connector

Use

SAP Query is a component of SAP's Web Application Server that allows you to create custom reports without any ABAP programming knowledge. The BI SAP Query Connector uses SAP Query to allow applications created with the BI Java SDK to access data from these SAP operational applications. This connector is fully compliant with the J2EE Connector Architecture (JCA).

You can also use the BI SAP Query Connector to make these data sources available in SAP BI Systems via UD Connect, or you can create systems in the portal based on this connector.

The BI SAP Query Connector implements the BI Java SDK's IBIRelational interface.

Prerequisites

- SAP ERP Systems including R/3 systems 4.6C or greater
- Systems supported by SAP JCo with JCo installed

Connector Properties

Refer to the table below for the required and optional properties to configure for your connector:

BI SAP Query Connector Properties

Property	Description	Examples
UserName	Data source user name. User must have at least read access to the data source.	(your user name)
Password	Data source password.	(your password)
Language	Two-letter abbreviation of language. Sets the logon language for your system, and also specifies the language of exceptions evoked on the BI Java SDK layer.	`EN`=English `DE`=German
QueryArea	Single-character value (char) specifying whether the SAP Query area is global (client-independent) or local (client-dependent). Default value is local.	`X`=global [blank]=local
Client	Client specified as a three-digit integer.	`000`
MessageServer	Hostname for message server.	`server1.mydomain.com`
ApplicationServer	Hostname for application server.	`server1.mydomain.com`
SystemNumber	Two-digit integer specifying	`00`

	the SAP system (R/3) instance.	
ServerGroup	Logon group for load balancing.	A string, such as `PUBLIC` or `SPACE`
R3Name	SAP system (R/3) name.	`B20`

For the set of connection properties required for your system, refer to the JCo documentation in your JCo download.

If you have trouble establishing a connection to a server that does not have a `PUBLIC` logon group, try removing the properties for `R3Name` and `ServerGroup`. See the JCo documentation on the SAP Service Marketplace (***service.sap.com/connectors***) for more connection troubleshooting information.

See also:

- To configure BI Java Connector properties in the SAP NetWeaver Application Server's Visual Administrator, see Configuring the BI Java Connectors [Page 592].
- To create a system on the portal using the BI SAP Query Connector, see Creating Systems [External] and Editing BI SAP Query System Properties [External].
- For information about the BI Java SDK and its connection architecture, refer to the ***index.html*** file inside of the SDK distribution package
- For more information about Sun's J2EE Connector Architecture, see ***http://java.sun.com/j2ee/connector/***
- For more information about SAP Query, refer to the following document: SAP Query [External]

4.3.1.4.8.5 I XMLA Connector

Use

Microsoft's XMLA (XML for Analysis) facilitates Web services-based, platform-independent access to OLAP providers. The BI XMLA Connector enables the exchange of analytical data between a client application and a data provider working over the Web, using a SOAP-based XML communication API. The XMLA Connector sends commands to an XMLA-compliant OLAP data source in order to retrieve the schema rowsets and obtain a result set.

The BI XMLA Connector allows you to connect applications built with the BI Java SDK to data sources such as Microsoft Analysis Services, Hyperion, MicroStrategy, MIS, and BW 3.x. This connector is fully compliant with the J2EE Connector Architecture (JCA).

You can also use the BI XMLA Connector to make these data sources available in SAP BI Systems via UD Connect, or you can create systems in the portal based on this connector.

The BI XMLA Connector implements the BI Java SDK's IBIOlap interface.

Prerequisites

XMLA-compliant OLAP data sources, such as SAP BI (BW) system release 3.0A or greater

Connector Properties

Refer to the table below for the required and optional properties to configure for your connector:

BI XMLA Connector Properties

Property	Description	Examples
UserName	Data source user name. User must have at least read access to the data source.	(your user name)
Password	Data source password.	(your password)
Language	Two-letter abbreviation of language. Determines logon language for SAP BI systems, and specifies the language of exceptions evoked on the BI Java SDK layer. *Optional*.	`EN`=English `DE`=German

URL	Server URL.
	To find the URL of an SAP BI XMLA provider:
	Start transaction **SE37** and execute the function module `RSBB_URL_PREFIX_GET`.
	Use the following values for the import parameters of the function module:
	`I_HANDLERCLASS = CL_RSR_MDX_SOAP_HANDLER`
	`I_PROTOCOL = HTTP`
	`I_MESSAGESERVER =`
	The URL takes the following form: `*/sap/bw/xml/soap/xmla`.
		SAP BI XMLA provider -- `http://[domain:port]/sap/bw/xml/soap/xmla`
		Microsoft Analysis Server -- `http://[domain:port]/isapi/msxisapi.dll`

DataSource	Data source info property.	`Local Analysis Server`
Statefulness	Statefulness support. *Optional*. Default is `false`.	`true` or `false`
Logonmethod	Specifies authentication mechanism (logon method). If using `SAPLOGONTICKET`, see the *For the BI XMLA Connector only* section in Configuring the BI Java Connectors [Page 592] for information on using Single Sign-On with the BI XMLA Connector.	`UIDPW` - Log on to your XMLA provider using User/Password. Use this method for XMLA-based access to non-SAP BI providers, or for User/Password access to SAP BI providers. This is the sole authentication method supported in the unmanaged environment. `SAPLOGONTICKET` - Log on to your XMLA provider using SAP Logon

		Ticket (supported by SAP BI XMLA providers). Use this property if you want to use the SAP Logon Ticket (SSO) to access BI systems.
Timeout	Maximum time to wait for the SOAP connection (in milliseconds). Default is `30000` (thirty seconds).	`60000` - to increase timeout to one minute

See also:

- To configure BI Java Connector properties in the SAP NetWeaver Application Server's Visual Administrator, see Configuring the BI Java Connectors [Page 592].
- To create a system on the portal using the BI XMLA Connector, see Creating Systems [External] and Editing BI XMLA System Properties [External].
- The BI XMLA Connector relies on the SAP NetWeaver Business Intelligence XML for Analysis interface. For more information, see XML for Analysis [External].
- For information about the BI Java SDK and its connection architecture, refer to the ***index.html*** file inside of the SDK distribution package
- For more information about Sun's J2EE Connector Architecture, see ***http://java.sun.com/j2ee/connector/***
- For more information about XMLA, see:
  - ***http://www.xmla.org/docs_pub.asp***
  - ***http://msdn.microsoft.com/library/default.asp?URL=/library/techart/XMLAnalysis.htm***

4.3.1.5 Transferring Data Using DB Connect

Purpose

By default, when the BI application server starts, SAP kernel opens a connection to the database on which the SAP system is running. In the remainder of this section, this connection is referred to as the (SAP) default connection. All SQL commands that are submitted by the SAP kernel or ABAP programs (irrespective of whether they are open or native SQL commands), automatically refer to this default connection; they run in the context of the database transaction that is active in this connection. Connection data, such as database user name, user password, or database name are taken either from the profile parameters or from the corresponding environment variables (this is database specific).

You use DB Connect to open other database connections in addition to the default connection and use these connections to transfer data into a BI system from tables or views.

See also:

DB Connect Architecture [Page 605]

Implementation Considerations

If you want to create a connection to an external database, you need relevant knowledge and experience of the source database in the following areas:

- Tools
- Database-specific SQL syntax
- Database-specific functions

You also need relevant knowledge of the source application so that you can transfer semantically utilizable data into BI.

If the BI DBMS and the source DBMS are different, you have to install a database-specific DB client for the respective source-database management system (DBMS) on the BI application server before you can use the DB Connect functions.

In all cases, you need to license the database-specific DB client with the database manufacturer. For information about the database-specific DB client, see the information from the respective database manufacturers.

In addition, the SAP-specific part of the database interface (the Database Shared Library (DBSL)) must be installed on the BI application server for the corresponding source database management system. For more information, see Installing the Database Shared Library (DBSL) [Page 606].

The information contained in the DB Connect documentation is subject to change. Always refer to the SAP Notes listed in the documentation.

Integration

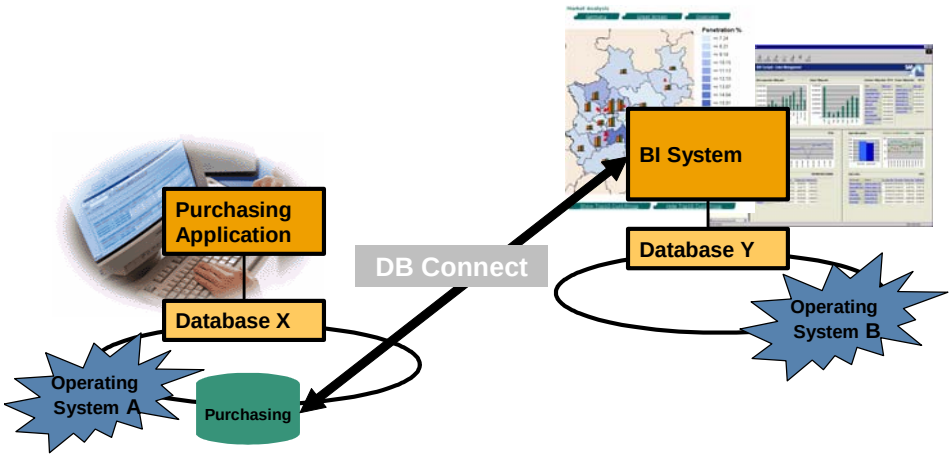
Using DB Connect, BI offers flexible options for extracting data directly into BI from tables and views in database management systems that are connected to BI using connections other than the default connection. You can use tables and views in database management systems that are supported by SAP to transfer data. You use DataSources to make the data known to BI. The data is processed in BI in the same way as data from all other sources.

Features

With DB Connect, you can load data into BI from a database system that is supported by SAP, by:

- Connecting a database to BI as a source system, thereby creating a direct point of access to external relational database management systems (RDBMS).
- Making metadata known to BI by generating a DataSource.

Example



A purchasing application runs on a system that is based on DBMS X. Before you can analyze the data in the purchasing application, you have to load the data into a BI system. The BI system is based on DBMS Y. DBMS Y can be the same as DBMS X or can be different to DBMS X. If DBMS Y is the same as DBMS X, you do not need to install the database-specific client and the database-specific DBSL. DB Connect allows you to connect the DBMS of the purchasing application and extract data from the database tables or views and transfer it into BI.

4.3.1.5.1 DB Connect Architecture

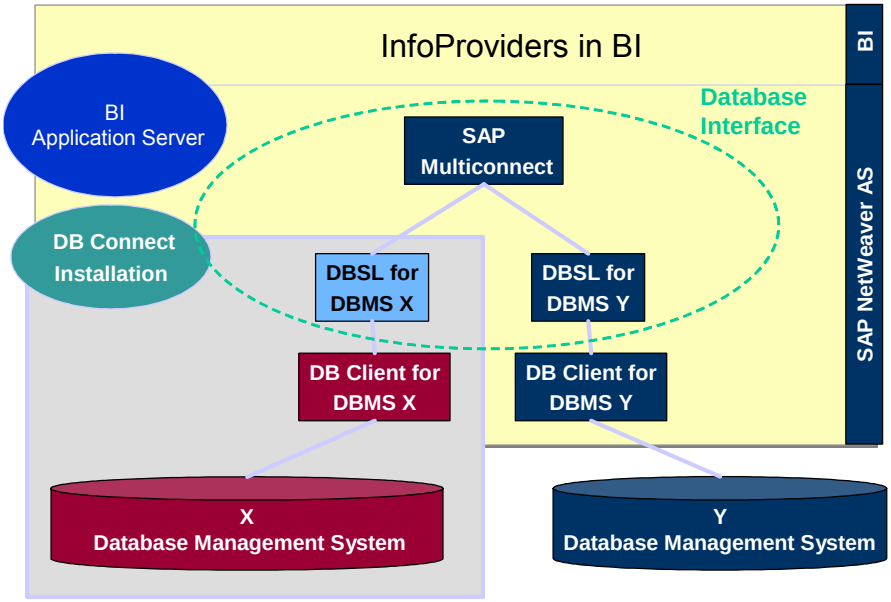
The multiconnect functions that are delivered as a SAP NetWeaver component allow you to open extra database connections in addition to the SAP default connection and use these connections to access external databases.

For more information, see SAP Note 323151 – *Multiple DB Connections with Native SQL*.

You can also use DB Connect to establish a connection of this type as a source system connection to BI. The DB Connect enhancements to the database interface allow you to transfer data straight into BI from the database tables or views of external applications.

For the default connection, DB Client and DBSL are preinstalled for the database management system. If you want to use DB Connect to transfer data into the BI system from other database management systems, you need to install both the database-specific DB Client and the database-specific DBSL on the BI application server that you are using to run DB connect.

In the following graphic, the BI system runs on DBMS Y. Therefore you do not need to install DBSL and DB Client for the source DBMS Y. However, if you want to load data from a DBMS X table or view, you have to install DBSL and DB Client for DBMS X.



4.3.1.5.2 Installing the Database Shared Library (DBSL)

Purpose

You find the database-dependent part of the SAP database interface in its own library. This is linked dynamically to the SAP kernel. This database library contains the Database Shared Library (DBSL) and libraries belonging to the corresponding database manufacturers. These are either statically or dynamically linked to the database library.

When you start an SAP system, the database-dependent database library is loaded before the DBSL is called for the first time. The system searches for the library in the directory indicated by the environment variable **DIR_LIBRARY** (for example,. /usr/sap/<SAPSID>/SYS/exe/run). The environment variable **dbms_type** contains the name of the required database management system. When the system is started, an attempt is made to load the library belonging to the required database management system from the directory that is indicated by the environment variable DIR_LIBRARY.

For more information about the database library, see SAP Note *400818 - Information about the R/3 Database Library*.

One of the advantages of this architecture is that a work process can include connections to several different databases belonging to different manufacturers.

To use DB Connect to transfer data into BI, you need to have installed the SAP-specific part of the database interface, the DBSL, for the corresponding source-database management system for each BI application server.

Process Flow

The database library is available in the SAP Service Marketplace in the SAR archives LIB_DBSL<xxx>.SAR, in the patch directories. These are not specific to the database manufacturers.

1. You access the required directory from the Software Center on SAP Service Marketplace at: *http://service.sap.com/swdc → Download → Support Packages and Patches → Entry by Application Group → SAP NetWeaver → SAP NetWeaver → <relevant SAP NetWeaver Release> → Entry by Component → Application Server ABAP → <relevant SAP Kernel> → <Operating system of the BI application server> → <source database management system> → LIB_DBSL**<xxx>.**SAR*.
2. Load the file into the directory indicated by the environment variable DIR_LIBRARY.

   The file LIB_DBSL.SAR forms a complete SAP kernel patch together with the database-independent DW.SAR archive. (SAP Note 19466 - *Downloading SAP Kernel Patches* describes how kernel patches are imported).
3. Next, unpack the SAR archive using the SAPCAR tool. Before doing so, refer to SAP Note 212876- *The New Archiving Tool* SAPCAR

   We recommend that you also download the latest DBSL from the SAP Service Marketplace for SAP DB and MaxDB databases.

Result

In the directory defined in the environment variable of the BI Application server, you find the library **db<dbs>slib.<ext>**, where <dbs> is the SAP-specific ID for the database management system and <ext> is the enhancement of the shared libraries in the respective operating system.

The database library for the Oracle database management system in Windows is called **dboraslib.dll**.

After you have installed the database-specific DB Client, you have fulfilled all installation prerequisites for using DB Connect. For information about the database-specific DB client, see the information from the respective database manufacturers.

4.3.1.5.3 Supported Databases

In general, BI application servers can only be supported for DB Connect on operating system versions where a SAP Database Shared Library (DBSL) has been released for the BI database and the source database.

The following information is subject to change. Always refer to the relevant SAP Notes.

Supported Database Management Systems	Database Source System Requirements	BI System Requirements	Further Information
SAP DB (ada) or Version 7.5	Versions: SAP DB 7.2.5	DB Client: SAP DB Client	**SAP**

MaxDB (sdb) or higher More information: Software Information [External]	Build 3 or higher	Version SAP DB 7.3.1 or higher	**Note 520647**
Microsoft SQL Server (mss)	Versions: MS SQL Server 7.0 and MS SQL Server 2000 Application server: Windows NT	Application server: Windows NT DB Client: MS SQL 7 or higher We recommend that you use MS SQL 2000s highest service pack (see SAP Note 62988 – Service Packs for MS SQL Server).	**SAP Note 512739**
Oracle (ora)	Versions: Oracle 8.1.7.3 or higher (clients are upwards compatible) The connection may also work with versions earlier than Oracle 8.1.7.3. However, Oracle does not support these versions.	DB Client: Oracle 8.1.7.3 or higher (delivered with SAP Web Application Server 6.20)	**SAP Note 518241**
IBM DB2/390 (db2)	Versions: DB2/390 V6 or higher Refer to SAP Note 81737 – DB2/390 APAR List.	DB Client (ICLI): from 6.20	**SAP Note 523552**
IBM DB2/400 (db4)	Versions: DB2/400 V4R5 or higher Both EBCDIC and ASCII data can be read.	Application server: Windows NT DB Client: IBM Client Access Express and XDA Release V5R1 or higher and minimum release of source DB	**SAP Note 523381**
IBM DB2 UDB (db6)	Versions: DB2 UDB for Unix and Windows V8.1 or higher Only use FixPaks allowed by SAP. These are described in SAP Note 200225 - DB6: Supported FixPaks in SAP BW.	DB Client: DB2 UDB or higher Run-time Client for Unix and Windows that is supported by the SAP kernel of the BI system that is used	**SAP Note 523622**

4.3.1.5.4 Requirements for Database Tables and Database Views

Table Names, View Names and Field Names

The naming conventions for the ABAP dictionary usually apply for table names and field names.

Make sure that you only use tables and views in the extraction whose technical names consist solely of upper case letters, numbers, and underscores (_). Problems may arise if you use other characters.

You can use database views to convert original names in table names to uppercase and to apply other conversions. For more information see Database Users and Database Schemas [Page 610]

Code Page and Sort Sequence for Source System

SAP kernel-based systems, like the BI system, work on the assumption that the database was created with code page cp850 and is using sort sequence ‘bin’. The source system configuration may be different from this. If the sort sequence is different, sample search (like) and area search (between, >,<) operations for character fields may return different results.

If you use multibyte code pages in the source system to store data for character records with more than 256 characters (special characters for Japanese (kanji and hiragana), Korean, Chinese, and so on) there is a risk that some of the characters may be corrupted.

When you create the DataSource, you can check the result of the extraction in the preview to determine whether this problem has occurred.

Since data conversion problems and unexpected sort results may arise if the database source system and BI do not use the same code page, we recommend that you use the same code page in both the database source system and in BI.

DB Data Types

As a rule, the only data types that can be supported are those that can be modeled on ABAP Dictionary data types.

When you use DB data types, refer to the database-specific SAP notes for DB Connect shown below.

You can use database views to convert data types, if necessary. For more information see Database Users and Database Schemas [Page 610] .

List of Database-Specific SAP Notes

If you are using an MSS database, refer to SAP Note 512739.

If you are using an Oracle database, refer to SAP Note 518241.

If you are using an SAP DB or MaxDB database, refer to SAP Note 520647.

If you are using an IBM DB2/390 database, refer to SAP Note 523552.

If you are using an IBM DB2/400 database, refer to SAP Note 523381.

If you are using an IBM DB2 UDB database, refer to SAP Note 523622.

4.3.1.5.5 Database Users and Database Schemas

When a database user is created in a database management system (DBMS), the system generates a database schema of the same name. This type of database schema is a collection of database objects where tables and views are managed. The schema and the objects belong to the user. You can assign read-write authorizations to other users for the schema, tables and views.

If you want to use DB Connect to establish a connection to a database source system, you need to create a user name and password in the DBMS. In the following example, this user is referred to as the BI user. You use the BI user to work in the database schema that was created with the name of the BI user. The tables and views containing the application data are stored in the DBMS, usually in an applications schema. Make sure that the BI user has read access to the tables and views in the application schema that are transferred into the BI system. The BI user can only extract the data extraction and preview the data from the DataSource maintenance if he or she has read permission.

Data Flow with Transformation

To extract data from a DBMS, you only need one BI user and thus only one source system connection to this DBMS. When defining the DataSource, you can limit the selection of the source data by specifying a database user. If you specify a database user (application) on tab page *Extraction* in the DataSource maintenance, those tables and views that belong to the specified database user and that lie in the schema of this database user are displayed for selection.

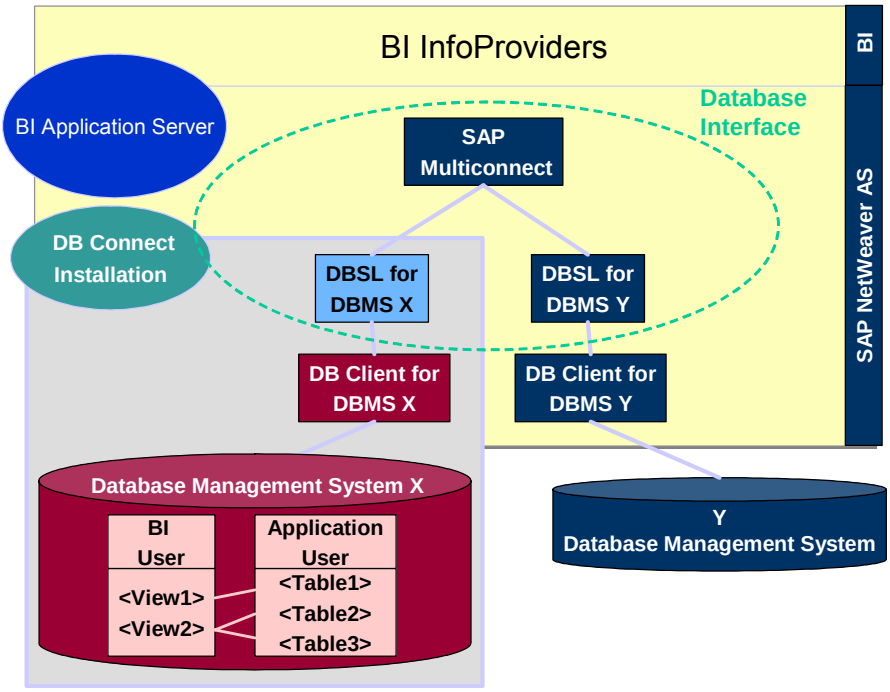
The tables and views that belong to the database user but that lie in a schema of a different database schema than the one specified are also displayed. The database user cannot extract these tables and views. In this case you can gain access to the data in the application schema using a view.

In some databases there might be schemas that do not correspond to any database user. If you would like to extract from a table of such a schema, you can give the BI user read permission for the table in this schema and create a view on the table in the schema of the BI user. You then define the DataSource fort he view in the schema of the BI user.

Further applications for views are described in the section below under points 3 and 4.

Data Flow with 3.x Objects

BI users who use a DataSource 3.x need permission to create views in their schema. You need these views in the schema of BI users in order to access tables and views in the application schema.



Using views, you can answer administration and authorization queries centrally in the source system.

1. To extract data from a DBMS, you only need one BI user and his or her schema, and thus only one source system connection to this DBMS. You use views in the BI user's schema to access the data that you want to extract that is stored in other schemas.

   If there are no views on data of the application schema in the schema of the BI user, you need an additional source system connection for which the database user is the BI user or connection user.

2. You can access tables with the same technical name by creating views with different names for these tables in the BI user's schema. In this way you can generate different DataSources for tables with the same name.

   If the tables contain similar semantic content, you can control the authorizations for the database user in such a way that he or she can only access the relevant tables.

3. You can structure the views in such a way that you are able to control access rights to the tables and restrict or reformat data as well as carry out join operations across several tables. Using views also makes it easier to localize errors.

   We recommend that if you need to perform conversions, you perform as many as possible in the view. This allows you to identify any errors or problems that arise at source-system level and you can resolve them as quickly as possible.

   You use conversion statements to

   - convert the names of database tables into capital letters

- convert dates from the internal date format used in the database to the SAP date format `YYYYMMDD`

4. By using views as an interface between a physical table and the BI system, you can use corresponding conversion statements in the view to make changes to the tables in the application schema, without affecting the view itself.

4.3.1.5.6 Creating Database Management Systems as Source Systems

Use

With DB Connect you have the option of opening extra database connections in addition to the SAP default connection. You use these connections during extraction to BI to access databases and transfer data into a BI system. To do this, you have to create a database source system in which the connection data is specified and made known to the ABAP runtime environment. The connection data is used to identify the source database and authenticate the database.

Prerequisites

- You have made the following settings in the Implementation Guide (IMG) under *SAP NetWeaver → Business Intelligence → Connections to Source Systems*:
  - General connection settings
  - Perform automatic workflow customizing
- As a rule, system changes are not permitted in productive systems. Connecting a system to BI as a source system, or connecting BI to a new source system, represents a change to the system. Therefore, you have to ensure that in the clients of the BI system that are affected, the following changes are permitted during the source system connection.
  - Cross-client Customizing and repository changes

    In the Implementation Guide (IMG) under *SAP NetWeaver → Business Intelligence → Links to Source Systems → General Connection Settings → Assign Logical System to Client*, select the relevant clients and choose *Goto → Details*. In the *Cross-Client Object Changes* field, choose the *Changes to Repository and Cross-Client Customizing Allowed* option.
  - Changes to the local developments and Business Information Warehouse software components

    You use transaction SE03 (Organizer Tools) to set the change options. Choose *Organizer Tools → Administration → Set Up System Change Option*. Choose *Execute*. On the next screen, make the following settings:.
  - Changes to the customer name range.

    Again, you use transaction SE03 to set the change option for the customer name range.
  - Changes to BI namespaces /BIC/ and /BI0/

    Again, use transaction SE03 to set the changeability of the BI namespace.
- If the source DBMS and BI DBMS are different:

  - You have installed the database-specific DB client software on your BI application server. You can get information about the database-specific DB client from the respective database manufacturers.
  - You have installed the database-specific DBSL on your BI application server [Page 606].

- In the database system, you have created a username and password that you want to use for the connection.

  See Database Users and Database Schemas [Page 610].

Procedure

Before you can open a database connection, all the connection data that is used to identify the source database and authenticate the database has to be made known to the ABAP runtime environment. For this, you need to specify the connection data for each of the database connections that you want to set up in addition to the SAP default connection.

1. In the source system tree in the Data Warehousing Workbench, choose *Create* in the context menu of the *DB Connect* folder.
2. On the following screen, specify the logical system name (= DB connection) and a descriptive text for the source system. Choose *Continue*.

   The *Change "Description of Database Connection" View: Detail* screen appears.
3. Select the database management system (DBMS) that you want to use to manage the database. This entry determines the database platform for the connection.
4. Under *User Name,* specify the database user under whose name you want the connection to be opened.
5. When establishing the connection, enter the user *DB Password* twice for authentication by the database. This password is encrypted and stored.
6. Under *Connection Info*, specify the technical information required to open the database connection. This information, which is needed when you establish a connection using NATIVE SQL, depends on the database platform and encompasses the database names and the database host on which the database runs. he string informs the client library of the database to which you want to establish the connection.

Connection information that depends on the database platform

Supported Database	CON_ENV Connection Information
SAP DB (ada) or MaxDB (dbs)	<server_name>-<db_name>
Microsoft SQL Server (mss)	MSSQL_SERVER=<server_name> MSSQL_DBNAME=<db_name> MSSQL_SERVER=10.17.34.80 MSSQL_DBNAME=Northwind (See SAP Note 178949 - MSSQL: Database MultiConnect with EXEC SQL)
Oracle (ora)	TNS Alias (See SAP Note 339092 - DB-MultiConnect with Oracle as a secondary database)

DB2/390 (db2)	 PORT=4730;SAPSYSTEMNAME=D6B;SSID=D6B0;SAPSYSTEM=71;SAPDBHOST=ihsapfc; ICLILIBRARY=/usr/sap/D6D/SYS/exe/run/ibmiclic.o The parameters describe the target system for the connection (see installation handbook DB2/390). The individual parameters (PORT=... SAPSYSTEMNAME=... .....) must be separated with **' '** , **','** or **';'**. (See SAP Note 160484 - DB2/390: Database MultiConnect with EXEC SQL)
DB2/400 (db4)	<parameter_1>=<value_1>;...;<parameter_n>=<value_n>; You can specify the following parameters: • AS4_HOST: Host name for the Remote DB Server. You have to enter the host name in the same format as is used under TCP/IP or OptiConnect, according the connection type you are using. You have to specify the AS4_HOST parameter. • AS4_DB_LIBRARY: Library that the DB server job needs to use as the current library on the remote DB server. You have to enter parameter AS4_DB_LIBRARY. • AS4_CON_TYPE: Connection type; permitted values are OPTICONNECT and SOCKETS. SOCKETS means that a connection is used using TCP/IP sockets. Parameter AS4_CON_TYPE is optional. If you do not enter a value for this parameter, the system uses connection type SOCKETS. For a connection to the remote DB server as0001 on the RMTLIB library using TCP/IP sockets, you have to enter: **AS4_HOST=as0001;AS4_DB_LIBRARY=RMTLIB;AS4_CON_TYPE=SOCKETS;** The syntax must be exactly as described above. You cannot have any additional blank spaces between the entries and each entry has to end with a semicolon. Only the optional parameter AS4_CON_TYPE=SOCKETS can be omitted. (See SAP Note 146624 - AS/400: Database MultiConnect with EXEC SQL) (For DB MultiConnect from Windows AS to iSeries, see Note 445872)
DB2 UDB (db6)	**DB6_DB_NAME=<db_name>** , where **<db_name>** is the name of the DB2 UDB database on which you want to run Connect. You want to establish a connection to the ‘mydb’ database. Enter **DB6_DB_NAME=mydb** as the connection information. (See SAP Note 200164 - DB6: Database MultiConnect with EXEC SQL)

7. Specify whether your database connection needs to be permanent or not.

   If you set this indicator, losing an open database connection (for example due to a breakdown in the database itself or in the database connection [network]) has a negative impact.

   Regardless of whether this indicator is set, the SAP work process tries to reinstate the lost connection. If this fails, the system responds as follows:

c. The database connection is **not permanent**, which means that the indicator is **not** set:

   The system ignores the connection failure and starts the requested transaction. However, if this transaction accesses the connection that is no longer available, the transaction terminates.

d. The database connection is **permanent**, which means that the indicator is set:

   After the connection terminates for the first time, each transaction is checked to see if the connection can be reinstated. If this is not possible, the transaction is not started – independently of whether the current transaction would access this special connection or not. The SAP system can only be used again once all the permanent DB connections have been reestablished.

   We recommend setting the indicator if an open DB connection is essential or if it is accessed often.

8. Save your entry and go back.
9. The *Change "Description of Database Connections" View: Overview* screen appears. The system displays the entry for your database connection in the table.
10. Go back.

Result

You have created IDoc basic types, port descriptions, and partner agreements. When you use the destinations that you have created, the ALE settings that enable a BI system to communicate with a database source system are created in BI in the background. In addition, the BI settings for the new connection are created in the BI system and the access paths from the BI system to the database are stored.

You have now successfully created a connection to a database source system. The system displays the corresponding entry in the source system tree. You can now create DataSources for this source system.

4.3.1.5.7 Creating DataSources for DB Connect

Use

Before you can transfer data from a database source system, the metadata (the table, view and field information) must be available in BI in the form of a DataSource.

Prerequisites

See Requirements for Database Tables or Views [Page 609]

You have connected a DB Connect source system.

Procedure

You are in the Data Warehousing Workbench in the DataSource tree.

1. Select the application components in which you want to create the DataSource and choose *Create DataSource*.

2. On the next screen, enter a technical name for the DataSource, select the type of DataSource and choose *Copy*.

   The DataSource maintenance screen appears.

3. Go to the *General* tab page.

   a. Enter descriptions for the DataSource (short, medium, long).

   b. As required, specify whether the DataSource builds an initial non-cumulative and can return duplicate data records within a request.

4. Go to the *Extraction* tab page.

   a. Define the delta process for the DataSource.

   b. Specify whether you want the DataSource to support direct access to data.

   c. The system displays *Database Table* as the adapter for the DataSource.

   Choose *Properties* if you want to display the general adapter properties.

   d. Select the source from which you want to transfer data.

   - Application data is assigned to a database user in the Database Management System (DBMS). You can specify a database user here. In this way you can select a table or view that is in the schema of this database user. To perform an extraction, the database user used for the connection to BI (also called BI user) needs read permission in the schema of the database user.

     If you do not specify the database user, the tables and views of the BI user are offered for selection.

   - Call the value help for field *Table/View*.

     In the next screen, select whether tables and/or views should be displayed for selection and enter the necessary data for the selection under *Table/View*. Choose *Execute*.

   - The database connection is established and the database tables are read. The *Select DB Object Names* screen appears. The tables and views belonging to the specified database user that correspond to your selections are displayed on this screen. The technical name, type and database schema for a table or view are displayed.

     Only use tables and views in the extraction whose technical names consist solely of upper case letters, numbers, and underscores (_). Problems may arise if you use other characters.

     Extraction and preview are only possible if the database user used in the connection (BI user) has read permission for the selected table or view.

     Some of the tables and views belonging to a database user might not lie in the schema of the user. If the responsible database user for the selected table or view does not match the schema, you cannot extract any data or call up a preview. In this case, make sure that the extraction is possible by using a suitable view. For more information, see Database Users and Database Schemas [Page 610].

5. Go to the *Proposal* tab page.

   The fields of the table or view are displayed here. The overview of the database fields tells you which fields are key fields, the length of the field in the database compared with the length of the field in the ABAP data dictionary, and the field type in the

database and the field type in the ABAP dictionary. It also gives you additional information to help you check the consistency of your data.

A proposal for creating the DataSource field list is also created. Based on the field properties in the database, a field name and properties are proposed for the DataSource. Conversions such as from lowercase to uppercase or from " " (space) to "_" (underlining) are carried out. You can also change names and other properties of the DataSource field. Type changes are necessary, for example, if a suitable data type is not proposed. Changes to the name could be necessary if the first 16 places of field names on the database are identical. The field name in the DataSource is truncated after 16 places, so that a field name could occur more than once in proposals for the DataSource.

When you use data types, be aware of database-specific features. For more information, see Requirements for Database Tables and Views [Page 609].

6. Choose *Copy to Field List* to select the fields that you want to transfer to the field list for the DataSource. All fields are selected by default.
7. Go to the *Fields* tab page.

   Here you edit the fields that you transferred to the field list of the DataSource from the *Proposal* tab page.

   a. Under *Transfer*, specify the decision-relevant DataSource fields that should be available for extraction and transferred to BI.
   b. If required, change the values for the key fields of the source.

      These fields are generated as a secondary index in the PSA. This is important in ensuring good performance for data transfer process selections, in particular with semantic grouping.
   c. Specify whether the source provides the data in the internal or external format.
   d. If you choose an *External Format*, ensure that the output length of the field (external length) is correct. Change the entries, as required.
   e. If required, specify a conversion routine that converts data from an external format into an internal format.
   f. Select the fields for which you want to be able to set selection criteria when you schedule a data request using an InfoPackage. Data for this type of field is transferred in accordance with the selection criteria specified in the InfoPackage.
   g. Choose the selection options (such as EQ, BT) that you want to be available for selection in the InfoPackage.
   h. Under *Field Type*, specify whether the data to be selected is language-dependent or time-dependent, as required.
8. Check the DataSource.

   The field names are checked for upper and lower case letters, special characters, and field length. The system also checks whether an assignment to an ABAP data type is available for the fields.
9. Save and activate the DataSource.
10. Go to the *Preview* tab page.

    If you choose *Read Preview Data*, the specified number of data records, corresponding to your field selection, is displayed in a preview.

This function allows you to check whether the data formats and data are correct. If you can see in the preview that the data is incorrect, try to localize the error.

See also: Localizing Errors [Page 618]

Result

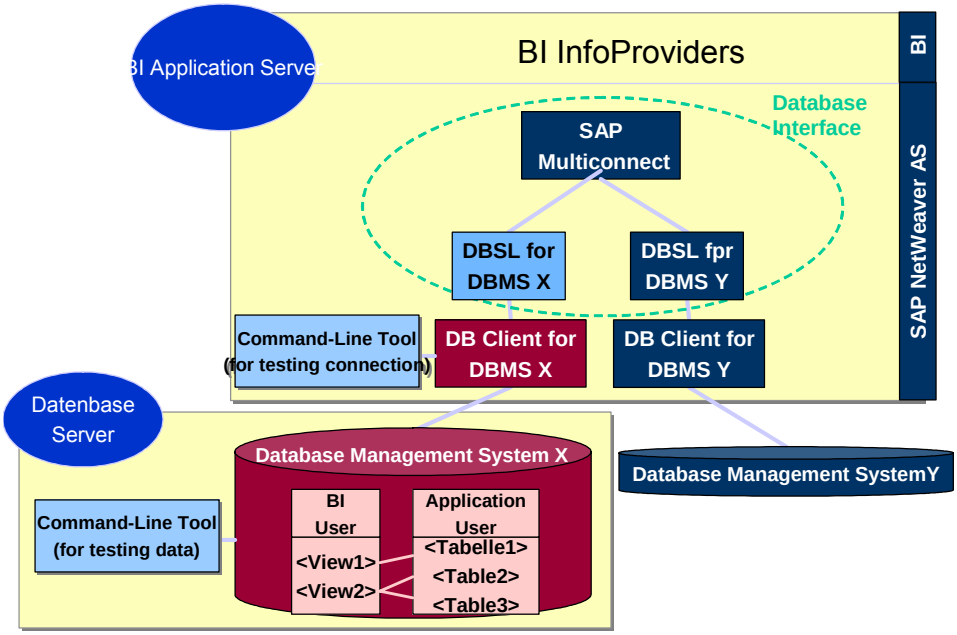
The DataSource is created and is visible in the Data Warehousing Workbench in the DataSource overview for the database source system under the application component. When you activate the DataSource, the system generates a PSA table and a transfer program.

You can now create an InfoPackage. You define the selections for the data request in the InfoPackage. The data can be loaded into the entry layer of the BI system, the PSA. Alternatively you can access the data directly if the DataSource supports direct access and you have a VirtualProvider in the definition of the data flow.

4.3.1.5.8 Localizing Errors

The preview function that is available when you process a DataSource enables you to identify potential problems before you actually load the data. If you notice in the preview that the data is incorrect, the following options are available to help you localize the error:

1. In the database management system (DBMS), use a SELECT command for a view to check which data is going to be delivered. Using a command-line tool on the database server, for example SQLPLUS for Oracle or db2 for IBM DB2/390, you can use this same SELECT command to test whether the data that has been read is correct. If you find an error, fix it in the DBMS.
2. If the error is not in the DBMS, use one of the command-line tools on the BI application server to establish a connection to the DBMS as a BI user. Use the SELECT command to test whether the DB client on the BI application server can see the data and whether this data is correct. If the DB client cannot see the data, it is likely that there is a connection error.
3. If the DB client on the BI application server can see the data and the data is correct, the error is in the BI system.



4.3.1.5.9 Updating Metadata

During a data load, the system identifies any inconsistencies between the metadata in the database source system and the metadata in BI.

You update metadata for DB Connect DataSources manually in the BI system. Proceed as when creating DataSources [Page 615] or generating DataSources [Page 619].

4.3.1.5.10 sing 3.x DataSources

Use

Before you can transfer data from a database source system, the metadata (the table, view and field information) must be available in BI in the form of a DataSource.

If your dataflow is modeled with objects based on the old concept (3.x InfoSource, 3.x transfer rules, 3.x update rules), you can use a 3.x DataSource to transfer data into BI from a database source system.

You can use a 3.x DataSource with restrictions in a data flow with transformation. This emulation can be used to prepare a migration of the 3.x DataSource.

Prerequisites

See Requirements for Database Tables or Views [Page 609].

Generating 3.x DataSources

In the context menu of a database source system, choose *Additional Functions* → *Select Database Tables* to generate a 3.x DataSource for database source systems. First, you

choose a selection of tables for a database source system and create a connection to the database source system. Next, you select the table fields for a specific table of the database source system and specify whether you want these table fields to be available for selection in BI. Finally, you generate the 3.x DataSource. The DataSource includes the set of fields that you want the system to read from the database source system during extraction.

You are on the *DB Connect: Overview of Tables and Views* screen.

1. In the first step, you select a table or view catalog from a database source system.

   a. Select the database source system from which you want to transfer data. The database source system or database connection is uniquely identified by the name of the logical system.

   b. Specify which tables or views you want to be displayed for selection.

      We recommend that you use the views in the schema of the database user in the Database Management System (DBMS) to access the tables and views containing application data.

      For more information, see Database Users and Database Schemas [Page 610].

   c. Specify whether you want tables or views to be displayed for selection.

   d. Choose *Execute*.

      The database connection is established and the database tables are read.

      The *DB Connect: Overview of Tables and Views* screen appears. On this screen the system displays, in accordance with your selections, the tables and views that are stored in the database schema of the database user for which the connection has been established.

      The technical name, type, and database schema for a table or a view are displayed in the *Selection of Database Tables/Views*. The entry in field *Table Information* shows whether the table or view is available for extraction. The icon indicates that tables and views are not available for extraction. If a table or view has no entry in this field, it is available for extraction. The *DataSource Name* field tells you whether a DataSource has already been generated for a table or a view.

      - Make sure that you only use tables and views in the extraction whose technical names consist solely of upper case letters, numbers, and underscores (_). Problems may arise if you use other characters.
      - A DataSource whose technical name consists of the prefix **6DB**_ and the technical name of the table or view is generated from the table or view. Since the names for DataSources in BI are limited to 30 characters, the technical name of the database table or view can be no longer than 26 characters.
        Tables and views with longer technical names are therefore **not** available for extraction.

2. In the second step, you specify the table fields for the DataSource that you are going to generate.

   a. In the overview, select a table or a view and choose *Edit DataSource*.

      The *DB Connect: Select Fields* screen appears. The following is displayed:

- Information about the database
- Information about the DataSource that you are going to generate
- The fields for the table or view

b. For the DataSource that you are going to generate, specify the application component in the source system tree of the Data Warehousing Workbench under which you want to add the DataSource. For the database source system, this application component hierarchy corresponds to the hierarchy in the InfoSource tree. In the default settings, the DataSource is assigned to the NODESNOTCONNECTED (unassigned nodes) application component.

c. Select the DataSource type.

The overview of the database fields tells you which fields are key fields, the length of the field in the database compared with the length of the field in the ABAP data dictionary, and the field type in the database and the field type in the ABAP dictionary. It also gives you additional information to help you check the consistency of your data.

When you use data types, be aware of database-specific features. For more information, see the database-specific comments under Requirements for Database Tables and Views [Page 609].

d. Set the *Selection* indicator and select the table fields and view fields that you want to be available for extraction from this table or view.

- The entry in field *Information* tells you whether the field is available for extraction. The icon indicates fields that are not available for extraction. Table fields and views fields, for which there is no entry next to this field, are available for extraction.
- Note that technical field names can be no longer than 16 characters long and must contain solely upper case letters, numbers, and underscores (_). Problems may arise if you use other characters. You cannot use fields with reserved field names, such as COUNT. Fields that do not comply with these restrictions are **not** available for extraction.

e. Select the fields for which you want to be able to set selection criteria when you schedule a data request with an InfoPackage. Data for this type of field is transferred in accordance with the selection criteria specified in the InfoPackage.

We recommend that to improve system performance during extraction, you only make selections using key fields and fields for which the secondary index is X.

3. If you choose the *Display Table Contents* option, a maximum of 20 data records that correspond to your field selection are displayed in a preview screen.

   This function allows you to check whether the data formats and data are correct. If you can see in the preview that the data is incorrect, try to localize the error [Page 618].

4. Check the DataSource.

   The field names are checked for upper and lower case letters, special characters, and field length. The system also checks whether an assignment to an ABAP data type is available for the fields.

5. Generate the DataSource.

Result

The DataSource is generated. It is visible in the Data Warehousing Workbench in the DataSource overview for the database source system under the assigned application component.

After you have assigned the DataSource to an existing InfoSource or a new InfoSource, assigned the DataSource fields to InfoObjects and activated the transfer rules, you need to create an InfoPackage. In the InfoPackage, you define the selections for the data request.

You have to use the PSA to load the data.

You cannot use the delta update method with DB Connect. In this case, you can perform a delta request using the selections (time stamp, for example).

Using and Migrating Emulated 3.x DataSources

For more information, see Emulation, Migration and Restoring DataSources [Page 37].

4.3.1.6 Transferring Data from Flat Files

Purpose

BI supports the transfer of data from flat files, files in ASCII format (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange) or CSV format (**C**omma **S**eparated **V**alue). For example, if budget planning for a company's branch offices is done in Microsoft Excel, this planning data can be loaded into BI so that a plan-actual comparison can be performed. The data for the flat file can be transferred to BI from a workstation or from an application server.

Process Flow

1. You define a file source system.
2. You create a DataSource in BI, defining the metadata for your file in BI.
3. You create an InfoPackage that includes the parameters for data transfer to the PSA.

The metadata update takes place in DataSource maintenance of BI.

4.3.1.6.1 Creating DataSources for File Source Systems

Use

Before you can transfer data from a file source system, the metadata (the file and field information) must be available in BI in the form of a DataSource.

Prerequisites

Note the following with regard to CSV files:

- Fields that are not filled in a CSV file are filled with a blank space if they are character fields and with a zero (0) if they are numerical fields.

- If separators are used inconsistently in a CSV file, the incorrect separator (which is not defined in the DataSource) is read as a character and both fields are merged into one field and may be shortened. Subsequent fields are no longer in the correct order.

Note the following with regard to CSV files and ASCII files:

- The conversion routine that are used determine whether you have to specify leading zeros. See Conversion Routines in BI Systems [Page 98].
- For dates, you usually use the format YYYYMMDD, without internal separators. Depending on the conversion routine that is used, you can also use other formats.

Notes on Loading

When you load external data, you can load the data into BI from any workstation. For performance reasons, however, you should store the data on an application server and load it into BI from there. This means that you can also load the data in the background.

If you want to load a large amount of transaction data into BI from a flat file and you can specify the file type of the flat file, you should create the flat file as an ASCII file. From a performance point of view, loading data from an ASCII file is the most cost-effective method. Loading from a CSV file takes longer because in this case, the separator characters and escape characters have to be sent and interpreted. In some circumstances, generating an ASCII file may involve more effort.

Procedure

You are in the Data Warehousing Workbench in the DataSource tree.

1. Select the application components in which you want to create the DataSource and choose *Create DataSource*.
2. On the next screen, enter a technical name for the DataSource, select the type of DataSource and choose *Copy*.

   The DataSource maintenance screen appears.
3. Go to the *General* tab page.
   a. Enter descriptions for the DataSource (short, medium, long).
   b. As required, specify whether the DataSource builds an initial non-cumulative and can return duplicate data records within a request.

      Specify whether you want to generate the PSA for the DataSource in untyped format: If the PSA is not typed it is not generated in a typed structure but is generated with character-like fields of type CHAR only.

      Use this option if conversion during loading causes problems, for example, because there is no appropriate conversion routine, or if the source cannot guarantee that data is loaded with the correct data type.

      In this case, after you have activated the DataSource you can load data into the PSA and correct it there.
4. Go to the *Extraction* tab page.
   a. Define the delta method for the DataSource.
   b. Specify whether you want the DataSource to support direct access to data.
   c. Real-time data acquisition is not supported for data transfer from files.
   d. Select the adapter for the data transfer. You can load text files or binary files from your local work station or from the application server.

Text-type files only contain characters that can be displayed and read as text. CSV and ASCII files are examples of text files. For CSV files you have to specify a character that separates the individual field values. In BI you have to specify this separator character and an escape character which specifies this character as a component of the value, as required. After you have specified these characters you have to use them in the file. ASCII files contain data in a specified length. The defined field length in the file must be the same as the length of the assigned field in BI.

Binary files contain data in the form of Bytes. A file of this type can contain any type of Byte value, including Bytes that cannot be displayed or read as text. In this case, the field values in the file have to be the same as the internal format of the assigned field in BI.

Choose *Properties* if you want to display the general adapter properties.

e. Select the path to the file that you want to load or enter the name of the file directly, for example C:/Daten/US/Kosten97.csv.

You can also create a routine that determines the name of your file. If you do not create a routine to determine the name of the file, the system reads the file name directly from the *File Name* field.

f. Depending on the adapter and the file to be loaded, make further settings.

- **For binary files:**

  Specify the character record settings for the data that you want to transfer.

- **Text-type files:**

  Specify how many rows in your file are header rows and can therefore be ignored when the data is transferred.

  Specify the character record settings for the data that you want to transfer.

  For ASCII files:

  If you are loading data from an ASCII file, the data is requested with a fixed data record length.

  For CSV files:

  If you are loading data from an Excel CSV file, specify the *data separator* and the *escape character.*

  Specify the separator that your file uses to divide the fields in the *Data Separator* field.

  If the data separator character is a part of the value, the file indicates this by enclosing the value in particular start and end characters. You enter these start and end characters in the *Escape Characteristics* field.

  You chose the ; character as data separator. However, your file contains the value **12;45** for a field. If you set **"** as the escape character, the value in the file must be **12;45"** so that **12;45** is loaded into BI. The complete value that you want to transfer has to be enclosed by the escape characters.

  If the escape characters do not enclose the value but are used within the value, the system interprets the escape characters as a normal part of

the value. If you have specified “ as the escape character, the value 12”45 is transferred as 12”45 and 12”45” is transferred as 12”45”.

In a text editor (for example, note pad) check the escape character and separator currently being used in the file. These depend on the country version of the file you used.

Note that if you do not specify an escape character, the space character is interpreted as the escape character. We recommend that you use a different character as the escape character.

If you select the *Hex* indicator, you can specify the data separator and the escape character in hexadecimal format. When you enter a character for the data separator and the escape character, these are displayed as hexadecimal code after the entries have been checked. A two character entry for a data separator or an escape sign is always interpreted as a hexadecimal entry.

g. Make the settings for the number format (thousand separator and character used to represent a decimal point), as required.

h. Make the settings for currency conversion, as required.

i. Make any further settings that are dependent on your selection, as required.

5. Go to the *Proposal* tab page.

This tab page is only relevant for CSV files. For files in different formats, define the field list on the *Fields* tab page.

Here you create a proposal for the field list of the DataSource based on the sample data from your CSV file.

a. Specify the number of data records that you want to load and choose *Upload Sample Data*.

The data is displayed in the upper area of the tab page in the format of your file.

The system displays the proposal for the field list in the lower area of the tab page.

b. Choose *Copy to Field List* to select the fields that you want to transfer to the field list for the DataSource. All fields are selected by default.

6. Go to the *Fields* tab page.

Here you edit the fields that you transferred to the field list of the DataSource from the *Proposal* tab page. If you did not transfer the field list from a proposal, you can define the fields of the DataSource here.

a. To define a field, choose *Insert Row* and specify a field name.

b. Under *Transfer*, specify the decision-relevant DataSource fields that you want to be available for extraction and transferred to BI.

c. Instead of generating a proposal for the field list, you can enter InfoObjects to define the fields of the DataSource. Under *Template* InfoObject, specify InfoObjects for the fields in BI. This allows you to transfer the technical properties of the InfoObjects into the DataSource field.

Entering InfoObjects here does not equate to assigning them to DataSource fields. Assignments are made in the transformation. When you define the transformation, the system proposes the InfoObjects you entered here as InfoObjects that you might want to assign to a field.

d. Change the data type of the field as required.

e. Specify the key fields of the DataSource.

These fields are generated as a secondary index in the PSA. This is important in ensuring good performance for data transfer process selections, in particular with semantic grouping.

f. Specify whether lowercase is supported.

g. Specify whether the source provides the data in the internal or external format.

h. If you choose the external format, ensure that the output length of the field (external length) is correct. Change the entries, as required.

i. If required, specify a conversion routine that converts data from an external format into an internal format.

j. Select the fields for which you want to be able to set selection criteria when you schedule a data request using an InfoPackage. Data for this type of field is transferred in accordance with the selection criteria specified in the InfoPackage.

k. Choose the selection options (such as EQ, BT) that you want to be available for selection in the InfoPackage.

l. If required, in the field type, specify whether the data to be selected is language-dependent or time-dependent.

7. Check, save and activate the DataSource.

8. Go to the *Preview* tab page.

If you select *Read Preview Data*, the number of data records you specified in your field selection is displayed in a preview.

This function allows you to check whether the data formats and data are correct.

Result

The DataSource is created and is visible in the Data Warehousing Workbench in the DataSource overview for the file source system in the application component. When you activate the DataSource, the system generates a PSA table and a transfer program.

You can now create an InfoPackage. You define the selections for the data request in the InfoPackage. The data can be loaded into the inbound layer of the BI system, the PSA. Alternatively you can access the data directly if the DataSource supports direct access and you have defined a VirtualProvider in the data flow.

4.3.1.6.2 Using Emulated 3.x DataSources

Use

You can display an emulated 3.x DataSource in DataSource maintenance in BI. Changes are not possible in this display. In addition, you can use emulation to create the (new) data flow

for a 3.x DataSource with transformations, without having to migrate the existing data flow that is based on the 3.x DataSource.

We recommend that you use emulation before migrating the DataSource in order to model and test the functionality of the data flow with transformations, without changing or deleting the objects of the existing data flow. Note that use of the emulated Data Source in a data flow with transformations has an effect on the evaluation of the settings in the InfoPackage. We therefore recommend that you only use the emulation in a development or test system.

Constraints

An emulated 3.x DataSource does not support real-time data acquisition, using the data transfer process to access data directly, or loading data directly (without using the PSA).

Prerequisites

If you want to use transformations in the modeling of the data flow for the 3.x DataSource, the transfer rules and therefore the transfer structure must be activated for the 3.x DataSource. The PSA table to which the data is written is created when the transfer structure is activated.

Procedure

To display the emulated 3.x DataSource in DataSource maintenance, highlight the 3.x DataSource in the DataSource tree and choose *Display* from the context menu.

To create a data flow using transformations, highlight the 3.x DataSource in the DataSource tree and choose *Create Transformation* from the context menu. You also use the transformation to set the target of the data transferred from the PSA.

To permit a data transfer to the PSA and further updating of the data from the PSA to the InfoProvider, select the DataSource 3.x in the DataSource tree and choose *Create InfoPackage* or *Create Data Transfer Process* in the context menu. We recommend that you use the processes for data transfer to prepare for the migration of a data flow and not in the production system.

Result

If you defined and tested the data flow with transformations using the emulation, you can migrate the DataSource 3.x after a successful test.

4.3.1.6.3 Transferring Data from Flat Files (3.x)

Purpose

SAP BW supports the transfer of data from flat files, files in ASCII format (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange) or CSV format (**C**omma **S**eparated **V**alue). For example, if budget planning for a company's branch offices is done in Microsoft Excel, this planning data can be loaded into SAP BW so that a plan-actual comparison can be performed. The data for the flat file can be transferred to SAP BW from a workstation or from an application server.

Prerequisites

See Maintaining InfoSources (Flat Files) [Page 286]

Process Flow

Definition and updating of metadata, that is, the DataSource is done manually for flat files in SAP BW. You can find more information about this, as well as about creating InfoSources for flat files under:

- Flexibly Updating Data from Flat Files [Page 287]
- Updating Master Data from a Flat File [Page 290]
- Uploading Hierarchies from Flat Files [Page 291]

The structure of the flat file and the metadata (transfer structure of the DataSource) defined in SAP BW have to correspond to one another to enable correct data transfer. Make especially sure that the sequence of the InfoObjects corresponds to the sequence of the columns in the flat file.

The transfer of data to SAP BW takes place via a file interface. Determine the parameters for data transfer in an InfoPackage and schedule the data request. You can find more information under Maintaining InfoPackages [Page 343] → Procedure for Flat Files [Page 344].

For flat files, delta transfer in the case of flexible updating is supported. You can establish if and which delta processes are supported during maintenance of the transfer structure. With additive deltas, the extracted data is added in BW. DataSources with this delta process type can supply both ODS objects and InfoCubes with data. During transfer of the new status for modified records, the values are overwritten in BW. DataSources with this delta process type can write the data into ODS objects and master data tables. You can find additional information under InfoSources with Flexible Updating of Flat Files [Page 288].

4.3.1.6.3.1 Updating Metadata for Flat Files and External Systems

Updating the Metadata for External Systems

Technically, metadata from external systems can be defined or updated manually, or using Business Application Programming Interface functionality (BAPI functionality).

If you access the BAPI interface with a third-party tool, the extraction tool from the third party can read the metadata automatically from the source system without a request from SAP BW or it can define the metadata in the third-party tool. Then the tool can transfer the metadata to SAP BW using the BAPI interface.

To manually change the metadata of an external system, enter the requested data in the transfer structure maintenance.

Updating the Metadata for Flat Files

You can only manually define and update the metadata for flat files. Enter the requested data in the transfer structure maintenance.

4.3.1.7 Data Transfer from External Systems

Purpose

In order to enable extraction of data and metadata from non-SAP sources on the application level, SAP BW provides open interfaces - staging BAPIs. BAPI [External]s (**B**usiness

Application **P**rogramming **I**nterface) are standardized programming interfaces that offer external access to the business processes and data of a SAP system. These interfaces enable connection between various third-party tools (such as **E**xtraction, **T**ransformation and **L**oading) and SAP BW. In this way, for example, data from an Oracle application can be transferred to SAP BW and can be evaluated there.

Process

The metadata can be defined or updated manually in the transfer structure maintenance in SAP BW. If you access BAPIs with a third-party tool, this tool can also automatically read the metadata from the source system without a request from SAP BW or it can define the metadata and then transfer it to SAP BW using BAPIs. SAP BW also offers interfaces with which third-party tools can create the metadata in the BW system.

Data transfer can take place via a data request from SAP BW or can be triggered by the third-party tool via BAPIs. The third-party tool loads the data from the external system and transforms it into the corresponding SAP BW format. Ensure that the structure of the transfer structure and the structure of the data structure for the extraction tool correspond to one another. Transformations for technical cleanup (such as date conversion) should already be implemented on the level of the extraction tool.

You can find more information on data transfer using staging BAPIs in your SAP BW system in the BAPI Explorer (transaction BAPI). On the *Hierarchical* tab page, choose *SAP Business Information Warehouse* → *Warehouse Management.*

See also:

Maintaining InfoSources (External System) [Page 300]

4.3.1.7.1 Updating Metadata for Flat Files and External Systems

Updating the Metadata for External Systems

Technically, metadata from external systems can be defined or updated manually, or using Business Application Programming Interface functionality (BAPI functionality).

If you access the BAPI interface with a third-party tool, the extraction tool from the third party can read the metadata automatically from the source system without a request from SAP BW or it can define the metadata in the third-party tool. Then the tool can transfer the metadata to SAP BW using the BAPI interface.

To manually change the metadata of an external system, enter the requested data in the transfer structure maintenance.

Updating the Metadata for Flat Files

You can only manually define and update the metadata for flat files. Enter the requested data in the transfer structure maintenance.

4.3.1.8 Notes on Data Transfer

The following section contains information about data transfer to a BI system. The information refers to special features regarding the type of data transfer and the data type.

4.3.1.8.1 oad Master Data to InfoProviders Straight from Source Systems

In data transfer process (DTP) maintenance, you can specify that data is not extracted from the PSA of the DataSource but is requested straight from the data source at DTP runtime. The *Do not extract from PSA but allow direct access to data source* indicator is displayed for the *Full* extraction mode if the source of the DTP is a DataSource. We recommend that you only use this indicator for small datasets; small sets of master data, in particular.

Extraction is based on synchronous direct access to the DataSource. The data is not displayed in a query, as is usual with direct access, but is updated straight to a data target without being saved in the PSA.

Dependencies

If you set this indicator, you do not require an InfoPackage to extract data from the source.

Note that if you are extracting data from a file source system, the data is available on the application server.

Using the *Direct Access* mode for extraction has the following implications, especially for SAP source systems (SAPI extraction):

- Data is extracted synchronously. This places a particular demand on the main memory, especially in the source system.
- The SAPI extractors may respond differently than during asynchronous load since they receive information by direct access.
- SAPI customer enhancements are not processed. Fields that have been added using the append technology of the DataSource remain empty. The exits RSAP0001, exit_saplrsap_001, exit_saplrsap_002, exit_saplrsap_004 do not run.
- If errors occur during processing in BI, you have to extract the data again since the PSA is not available as a buffer. This means that deltas are not possible.
- In the DTP, the filter only contains fields that the DataSource allows as selection fields. With an intermediary PSA, you can filter by any field in the DTP.

4.3.2 Real-Time Data Acquisition

Purpose

Real-time data acquisition supports tactical decision-making. It also supports operational reporting by allowing you to send data to the delta queue or PSA table in real-time. You then use a daemon to transfer DataStore objects to the operational DataStore layer at frequent regular intervals. The data is stored persistently in BI.

We recommend that you use real-time data acquisition if you want to transfer data to BI at shorter intervals (every minute) than scheduled data transfers and you need up-to-date data to be regularly available in reporting (several times a day, at least).

The following overview displays the differences between standard data acquisition using scheduled data requests and real-time data acquisition:

	Standard Data Acquisition	Real-Time Data Acquisition
Goal	Strategic Decision-Making (long-term planning)	Tactical Decision-Making (daily decisions)
Data Acquisition	Request Oriented (nightly background job)	Data Availability Oriented (permanently active background job)
Load Cycle	1/day ... 1/week	1/minute ... 1/hour
Resource Consumption	Processing normally nights (load balancing)	Permanent Resource Consumption

Prerequisites

- The DataSource has to support real-time data acquisition.

  Web service DataSources and DataSources from SAP source systems can support real-time data acquisition. In DataSource maintenance, the *Extraction* tab page shows whether the *Real Time* property is set.

  DataSources from SAP source systems can be used for real-time data acquisition if the following prerequisites are met:

  - BI Content DataSources have to be delivered with the property for supporting real-time data acquisition.
  - The *Real-Time-Enabl.* indicator has to be set in the generic delta settings for generic DataSources (see Delta Transfer to BI [Page 543]).

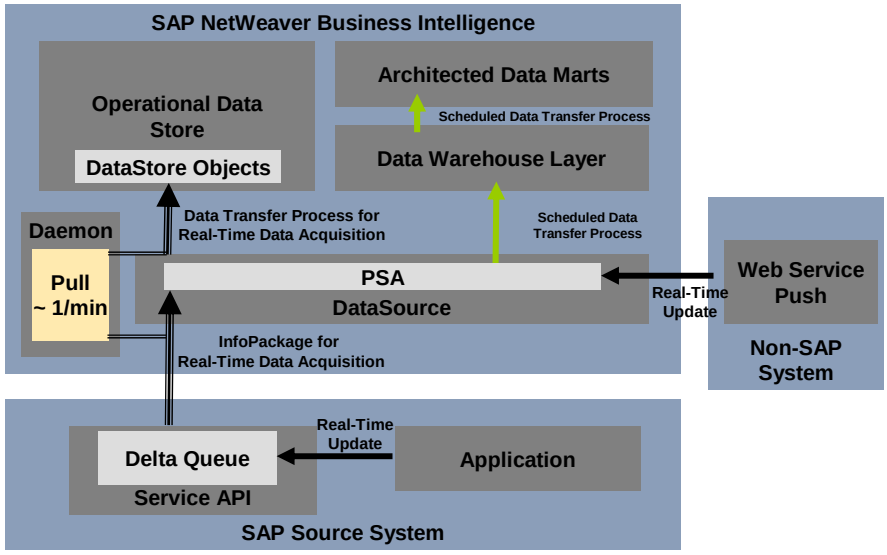
    Note that real-time data acquisition as a property of DataSources in SAP source systems is only possible technically under the following circumstances: In the source system, the BI Service API has the version Plug-In-Basis 2005.1 or for 4.6C source systems Plug-In 2004.1 SP10.

    Note also that you can only use a Web service DataSource for real-time data acquisition if you have defined it in the (new) DataSource maintenance in BI.

- You use a transformation between the PSA and DataStore object and update data into DataStore objects from the PSA using the data transfer process.

Process Flow

The following figure illustrates the process flow for real-time data acquisition:



Data is loaded into BI at frequent, regular intervals and is then posted to the DataStore objects that are available for operational reporting. In BI, special InfoPackages are used for this purpose and data transfer processes for real-time data acquisition are created to further process data from the PSA in the DataStore objects. This is scheduled and executed regularly by a dedicated background process (the daemon). Data is available for reporting as soon as it has been successfully posted to the DataStore object and activated. Refresh the query display to display the recent data. The query shows the time that the query was last updated by a daemon run, even if no new data was posted.

You can transfer data from the source to the entry layer of BI (the PSA) in two ways:

- Using a Web service

  You use the Web service to write the data from the source into the PSA. The transfer of data is controlled externally, without a request from BI. Only an InfoPackage (for full upload) is required to determine specific parameters for real-time data acquisition.

- Using a service API

  Data from an SAP source system can be loaded into the PSA using an InfoPackage created specifically for this purpose. This is triggered when the delta queue in the source system requests data. You have to simulate the initialization of the delta process for the DataSource beforehand.

  The following two scenarios are possible:

  - The source system application writes the data to the delta queue.

    In this case, the daemon retrieves the data without calling the extractor.

  - The application does not write data to the delta queue automatically; the extractor writes the data to the delta queue at the request of BI.

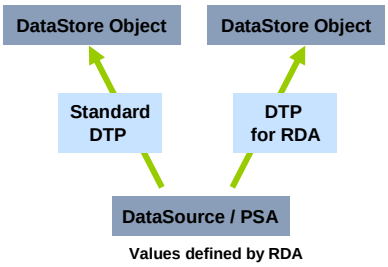
    For extractors that transfer data synchronously from BI to the service API on request (generic extractors, for example), the daemon calls the extractor, and the extractor writes the data to the delta queue. The data is transferred to BI directly from the delta queue.

Constraints

- You can only use real-time data acquisition to fill DataStore objects. A two-step data transfer is supported; data is first transferred into the PSA and then into the DataStore object. You cannot use the DataStore object as the source of a further real-time data transfer into another DataStore object. With real-time data acquisition, a data transfer that involves more than two steps is not possible.
- Master data cannot be transferred to the BI system with real-time data acquisition as otherwise the navigation attributes of the characteristic could no longer be used in aggregates. This is because aggregates cannot react to real-time updates since the change run in the RDA process cannot be triggered automatically for the loaded data.
- DataSources that are used for real-time data acquisition cannot be used for standard data transfer (scheduled staging). It is not possible to have multiple transfer mechanisms (RDA and scheduled data transfer) simultaneously, since the delta queue can only contain one entry for each DataSource and target system at any given time.

  In addition to a pure RDA update and a pure standard update, the following update methods, in which both transfer mechanisms are used in parallel, are possible for data that was transferred to the PSA by real-time data acquisition:

  The data is updated from the PSA to a DataStore object with a DTP for real-time data acquisition, and to another DataStore object with a standard DTP.



  Because of the different mechanisms when activating the DataStore object data in one DataStore object, you cannot load data simultaneously with real-time data acquisition and with a standard DTP in the delta process.

Converting an Existing Data Flow to Real-Time Data Acquisition

If you want to integrate the transfer of data with real-time data acquisition into an existing data flow, you have two options:

- Using two different DataSources

  One DataSource executes the standard data transfer. The other DataSource transfers the data with real-time data acquisition. The data is then combined in a MultiProvider.

- Using a single DataSource

  You have to replace the standard data transfer completely with a real-time data acquisition scenario.

More information: Real-Time Data Acquisition [External]

4.3.2.1 aemon for Real-Time Data Acquisition

Definition

Background process that processes the InfoPackages and data transfer processes assigned to it at regular intervals.

Use

The daemon controls and monitors the transfer process for real-time data acquisition. The daemon performs the following tasks depending on the scenario used:

- If you are transferring data from an SAP source system, the daemon starts the transfer of data into the PSA using an InfoPackage.
- It controls the status of the data transfer.
- It starts the further processing of data into the DataStore object using a data transfer process.
- It closes and opens requests when threshold values are reached.
- It triggers the subsequent process chains.

The daemon receives information from the InfoPackage and data transfer process about the data to be transferred or further updated, PSA tables or DataStore objects to be filled, and when a request should be closed and a new one opened.

Function

When you extract data, the daemon works on the basis of the list of DataSources assigned to it in the InfoPackage, extracts the data from the source systems, and transfers it to the PSA table and DataStore objects. It informs the service API in the source system when the data for the target system has been successfully updated. When the PSA request has been successfully closed and a new delta request has been opened, the updated data is deleted from the delta queue of the source system.

When you further update data from the PSA to the DataStore objects, the daemon works on the basis of the list of sources (DataSources) and targets (DataStore objects) assigned to it in the data transfer processes. It transfers data to the DataStore object. The data is directly activated and written to the change log.

Operating Mode

The daemon runs in a permanent background job and switches between extraction processes in idle mode. If you are using the service API to transfer data, the RFC connection to the source system remains open. This implies that a permanent RFC connection is required between each source system and BI for real-time data acquisition using the service API. To prevent the system from using too much main memory and to avoid keeping the RFC connection open for too long, the daemon stops independently on a regular basis and reschedules itself. As a result, the main memory can be released and a new RFC connection is opened. This happens without having to close the request for real-time data acquisition.

Error Behavior

The daemon writes each successfully executed step to a control table. If the extraction or update is terminated, the daemon restarts itself and continues the process from the point at which it was terminated. It repeats the entire substep with the granularity of a data package. For example, if execution of the DTP is terminated, the daemon gets the new data from the

source and loads it into the InfoProvider together with the data packages that terminated previously.
In the InfoPackage, you specify the maximum number of failed attempts before the process terminates (see also: Creating InfoPackages for Real-Time Data Acquisition [Page 647]).

4.3.2.2 Request Concept

To make it easier to process and report on data, the system synchronizes InfoPackage requests (PSA request), data transfer process requests (DTP request) and change log requests in the DataStore object and these requests remain open across several load processes.

Requests are opened and closed as follows:

1. When a daemon is started, it opens a PSA request in the first load process.
2. The system opens a DTP request and change log request when the PSA request contains data.
3. The daemon uses the threshold values defined in the InfoPackage to determine when to close a request. The data transfer process copies these settings from the InfoPackage.

   When a PSA request is closed, the related DTP and change log requests in the same load process are closed too.
4. When the requests are closed, new requests are opened automatically the next time the daemon accesses data and the data transfer for real-time data acquisition continues with these new requests.
5. Again, the system only opens a DTP request and change log request when the PSA request contains data.
6. If you are transferring data from an SAP source system, data is automatically deleted from the delta queue when the new PSA request is opened.

The data (and the open requests) are available for analysis and reporting as soon as they have been successfully updated and activated in the DataStore object. You can only update data from the DataStore object to a further InfoProvider (an InfoCube, for example), if the DTP (and change log) request is closed.

In some cases it is useful to close requests before the threshold value is reached. For more information, see:

Closing Requests Using Process Chains [Page 560]

Closing Requests in the Monitor for Real-Time Data Acquisition [Page 635]

4.3.2.2.1 Closing Requests in the Monitor for Real-Time Data Acquisition

Use

With real-time data acquisition, the requests (PSA requests, DTP requests and change log requests) remain open across several load processes. The requests are closed when the threshold values set in the InfoPackage are reached. The system opens new requests and

data transfer is continued using the new requests. In the monitor for real-time data acquisition, you can close PSA and DTP requests manually.

If errors occur, you can close an open request manually, correct the error and then restart data transfer.

Prerequisites

The daemon is inactive or has errors.

Procedure

In the context menu of the PSA or DTP request, choose *Close Request*.

Result

The selected request and any change log requests close.

4.3.2.3 ransferring Transaction Data from SAP Source Systems (RDA)

Use

If you want to use data for operational reporting that is available in SAP source systems, use the real-time data acquisition scenario.

Prerequisites

The DataSource is delta enabled and supports real-time data acquisition. For your source system, you have installed BI Service API Status SAP NetWeaver 2004s (Plug-In-Basis 2005.1) or higher, or for release 4.6C, Plug-In 2004.1 SP10.

Procedure

1. Activate the DataSource from BI Content and replicate the DataSource in the BI system, if necessary.

   See Installing BI Content DataSources in Active Version [Page 524].

   If no BI Content DataSource is available for your use case, you can create a generic DataSource and replicate it in the BI system. Set up the generic delta for the DataSource and set the real-time enabled indicator for the DataSource.

   See Maintaining Generic DataSources [Page 541].

2. Create a DataStore object.

   See Creating DataStore Objects [Page 155] .

3. Create a transformation with the DataSource as the source and the DataStore object as the target.

   See Creating Transformations [Page 242].

4. Create a data transfer process for real-time data acquisition with the DataSource as the source and the DataStore object as the target.

   See Creating Data Transfer Processes for Real-Time Data Acquisition [Page 649].

5. Create an InfoPackage for the DataSource for simulating the delta initialization and perform delta initialization.

   See Creating InfoPackages [Page 337].

6. Perform delta initialization.

   If you have transferred data during delta initialization, execute the following steps. Otherwise, proceed to step 9.

7. Update the data from the PSA into the DataStore object.

   a. Go to the maintenance screen for the data transfer process that you created in step 4.

   b. Choose *Change to Standard DTP.*

   c. On the *Execute* tab page, start the data transfer process.

   d. When the data has been successfully loaded into the DataStore object, activate the DTP request in the DataStore object.

8. Change the type of the data transfer process again.

   In the toolbar for data transfer process maintenance, choose *Change to Real-Time DTP*.

9. Create an InfoPackage for the DataSource for real-time data acquisition.

   See Creating InfoPackages for Real-Time Data Acquisition [Page 647].

10. Switch from the InfoPackage to the monitor for real-time data acquisition.

    In InfoPackage maintenance, on the *Schedule* tab page, choose *Assign.*

    The *Monitor for Real-Time Data Acquisition* appears (transaction RSRDA).

11. Define a daemon (using *Create Daemon*) or select an existing daemon.

12. Assign the DataSource (and the InfoPackage) to the daemon.

    The InfoPackage appears in the monitor for real-time data acquisition; the system displays the InfoPackage in the *Unassigned Nodes* area, under the DataSource for which the InfoPackage was created. In the context menu of the DataSource, choose *Assign Daemon* to assign the InfoPackage to a daemon. The daemon can now use the InfoPackage for data extraction.

13. Assign the data transfer process to the daemon.

    In the context menu of the DataSource, choose *Assign DTP.* The daemon uses the data transfer process to process data.

    You can update the data from the DataSource into multiple DataStore objects. In this case, assign the corresponding data transfer process to the daemon (using the DataSource).

14. In the monitor for real-time data acquisition, start the daemon.

    In the context menu of the daemon, choose *Start Daemon with All InfoPackages*.

    The daemon waits for a free background job.

Result

While the background job is running, the daemon starts the delta transfer using the InfoPackage for real-time data acquisition. The daemon executes three steps:

1. In the first step, the daemon calls the service API in the source system. The service API transfers the data records into BI (into the delta queue, where necessary) and passes them to the daemon. The data is updated into the PSA table.
2. In the second step, when the data has been successfully updated to the PSA table, the daemon confirms the transfer, and the service API changes the status of the records in the delta queue.

   Once the request has been successfully closed and the next request is open, the records are deleted from the delta queue.
3. In the third step, the daemon starts the data transfer process(es) for real-time data acquisition. This updates the data from the PSA table into the DataStore object. Changes are recorded in the appropriate change log request. The request that transfers the data to the PSA table (PSA request), the data transfer process request, and the change log request for each DataStore object have a 1:1 ratio. The data is activated automatically and written to the change log. This allows you to skip an additional activation step in the DataStore object; data is not written to the activation queue of the DataStore object.

The data is now available for reporting and analysis purposes.

4.3.2.4 ransferring Transaction Data Using Web Services (RDA)

Use

If you want to use data that is not available in SAP source systems for operational reporting, use a Web service to transfer the data into the BI system.

Procedure

1. Create a Web service DataSource.

   See Creating DataSources for Web Services [Page 558].
2. Implement the Web service in your application.
3. Create a DataStore object.

   See Creating DataStore Objects [Page 155].
4. Create a transformation with the DataSource as the source and the DataStore object as the target.

   See Creating Transformations [Page 242].
5. Create a data transfer process (DTP) for real-time data acquisition with the DataSource as the source and the DataStore object as the target.

   See Creating Data Transfer Processes for Real-Time Data Acquisition [Page 649].
6. Create an InfoPackage for the DataSource for real-time data acquisition.

   See Creating InfoPackages for Real-Time Data Acquisition [Page 647].

PSA requests for Web services remain open across several load processes. When you transfer data using Web services, you use this type of InfoPackage to define the size of the request or the time lapsed before the request is closed. The system checks the threshold values before it uses the request to update data. When a threshold value is reached, the system closes the current request and the data transfer is continued using a new request.

7. Switch from the InfoPackage to the monitor for real-time data acquisition.

   In InfoPackage maintenance, on the *Schedule* tab page, choose *Assign.*

   The *Monitor for Real-Time Data Acquisition* appears (transaction RSRDA).

8. Define a daemon (using *Create Daemon*) or select an existing daemon.
9. Assign the DataSource (and the InfoPackage) to the daemon.

   The InfoPackage appears in the monitor for real-time data acquisition; the system displays the InfoPackage in the *Unassigned Nodes* area, under the DataSource for which the InfoPackage was created. In the context menu of the DataSource, choose *Assign Daemon* to assign the InfoPackage to a daemon. The daemon can now use the InfoPackage for data extraction.

10. Assign the data transfer process to the daemon.

    In the context menu of the DataSource, choose *Assign DTP.* The daemon uses the data transfer process to process data.

    You can update the data from the DataSource into multiple DataStore objects. In this case, assign the corresponding data transfer process to the daemon (using the DataSource).

11. In the monitor for real-time data acquisition, start the daemon.

    In the context menu of the daemon, choose *Start Daemon with All InfoPackages*.

    The daemon waits for a free background job.

Result

When the Web service returns data to the BI system, it is updated into the PSA table.

While the background job is running, the daemon checks the PSA at regular intervals.

When the data has been successfully updated into the PSA table, the daemon starts the data transfer process(es) for real-time data acquisition. This updates the data from the PSA table into the DataStore object. Changes are recorded in the appropriate change log request. The request that transfers the data to the PSA table (PSA request), the data transfer process request, and the change log request for each DataStore object have a 1:1 ratio. The data is activated automatically and written to the change log. This allows you to skip an additional activation step in the DataStore object; data is not written to the activation queue of the DataStore object.

The data is now available for reporting.

Note

If you are using a Web service DataSource, you can use a standard DTP to update data from the PSA. You can also transfer master data to the BI system with a Web service in this way. More information: Transferring Data Using Web Services (Standard) [Page 559].

4.3.2.5 Monitor for Real-Time Data Acquisition

Use

In the monitor for real-time data acquisition (transaction RSRDA), the daemons currently available in the system are displayed in a tree structure along with the objects assigned to them.

The monitor displays the status of the daemons and the assigned objects. This enables you to monitor the daemon and the progress of the data update into the PSA and the DataStore object. You can execute various functions for an object in the context menu of a node.

You access the monitor by choosing *Monitors* in the *Administration* functional area of the Data Warehousing Workbench.

Features

Overview of Daemons

The system displays the DataSource that you have assigned to each daemon. For each DataSource, the system displays the relevant InfoPackage for real-time data acquisition that contains the parameters for the data transfer. Under *Unassigned Nodes,* the system displays any DataSources to which you have not yet assigned daemons. If you assigned a data transfer process to the DataSource, the system displays this on the same level as the InfoPackage. Under an InfoPackage or data transfer process, the system displays the open request that posts data to the PSA table or DataStore object. The system only displays the DTP request if the PSA contains data. If you have used the data transfer process to assign a subsequent process chain to a daemon, the system displays the process chain under the data transfer process.

For information about the navigation options in the daemon overview and the icons for the various overview levels, choose *Legend*.

You can trigger the system to check the background jobs and refresh the object display by choosing *Extras* → *Refresh*.

Daemon Functions

Function	Context Menu	Menu Path	Description/More Information
Create daemon	-	*Daemon* → *Create*	Enter an ID for the daemon. There can be up to 99 daemons assigned the numbers 01 to 99. Enter a description for the daemon. Specify the intervals (in minutes) at which the daemon is to access the data in the source and in the PSA.
Delete daemon	*Delete Daemon*	*Daemon* → *Delete*	When you delete a daemon, the objects assigned to this daemon are moved to *Unassigned Nodes*.
Display daemon settings	*Daemon Settings*	*Daemon* → *Info*	The system displays the daemon ID, the description of the daemon, and the interval (in minutes) at which the

			daemon accesses the data in the source and PSA. You can change the description and the time (in minutes) here.
Display log for the daemon	*Display Log*	*Daemon → Display Log*	The log entries for the daemon are displayed.
Display runtime information for daemon	*Runtime Info* Or double-click	-	In addition to the daemon ID and description, the following information is also displayed: • Job name and number of the job that executes the daemon • Server on which the daemon is running • Time stamp for the start of the job • Time stamp for the start of the daemon Since the daemon closes regularly and starts again in a new background job, the time stamps for the starting the daemon for starting the job are different. • User who started the daemon
Start daemon	Context menu of an inactive daemon: *Start Daemon with All InfoPackages*	-	Select a server or a server group. The daemon is scheduled on this server or on a server in the server group. If you do not specify a server, the system selects any available server. The daemon waits for a free background job. If the background job for the daemon is running, the daemon starts the data transfer for the InfoPackages and data transfer processes assigned to it according to the settings.
Stop upload	*Stop Upload*	*Daemon → Stop*	The data transfer for the assigned InfoPackages and DTPs is ended. The system then closes the requests (PSA, DTP and change log requests). The daemon displays (Stop) as the status if there are still open requests. The status of the daemon does not change to (inactive) until all requests are closed.

Restart daemon with errors	Context menu of a daemon that was terminated: *Restart Batch Job*	-	If you have eliminated the source of the error, you can restart the daemon. Select a server or a server group. The daemon is scheduled on this server or on a server in the server group. If you do not specify a server, the system selects any available server. The daemon waits until a background job is available and then restarts the data transfer with this job.

DataSource Functions

Function	Context Menu	Menu Path	Description/More Information
Start upload	*Start Upload*		The daemon executes the InfoPackage and the data transfer process for this DataSource during the next load process. Start the data transfer process in this way: • if you assigned the DataSource to a daemon that is already active • if you want to reactivate the data transfer process after an error occurs, the daemon is active, and the data transfer is running for other assigned DataSources
Stop upload	*Stop Upload*		The data transfer for this DataSource is closed and the requests (PSA, DTP, and change log request) are closed. This allows you to interrupt the data transfer if, for example, you need to adjust the data flow to reflect changed requirements.
Display DataSource	*DataSource Maintenance* *Or double-click*		The display mode of DataSource maintenance appears.
Assign InfoPackage and data transfer process to another daemon	*Assign Daemon*		Enter the ID of the daemon. An InfoProvider (and the related DTP) can only be assigned to **one** daemon. Therefore different DataSources that write data to the same InfoProvider always have to be

			assigned to the same daemon.
Delete assignment to a daemon	*Delete Assignment*		The DataSource and its related InfoPackages and data transfer processes are moved to *Unassigned Nodes*.
Assign data transfer process	*Assign DTP*		To assign the DTP to the DataSource, enter the technical name of the DTP. If there is only one DTP, the system makes the assignment automatically when you execute this function.

Functions for the InfoPackage

Function	Context Menu	Menu Path	Description/More Information
Call InfoPackage maintenance	*Scheduler* Or double-click	*InfoPackage* → *Maintain*	Note that you cannot call InfoPackage maintenance while the daemon is active.
Call monitor for InfoPackage	*Monitor*	*InfoPackage* → *Monitor*	The system displays the monitor with the PSA requests for the current and previous day.

Functions for the PSA Request

Function	Context Menu	Menu Path	Description/More Information
Call request monitor	*Monitor*	-	The system displays the monitor for the selected request.
Close request	*Close Request*	-	You can close a request if the daemon has terminated. See Closing Requests in the Monitor for Real-Time Data Acquisition [Page 635].

Data Transfer Process Functions

Function	Context Menu	Menu Path	Description/More Information
Call data transfer process maintenance	*DTP Maintenance*	-	-
Delete assignment to DataSource	*Delete Assignment*		The system no longer displays the data transfer process in the monitor for real-time data acquisition.
Call administration for DataStore object	*Manage InfoProviders* *Or double-click*	-	-
Assign process chains	*Assign*	-	In the dialog box, select a suitable

that follow the data transfer process to the daemon	*Subsequent Process Chain*		process chain. You can only select active process chains. Note that the subsequent process chain has to include a start process with scheduling option *Start Using API*. The process chain starts when the daemon closes a request. In this way, you can link further processing and update processes to real-time data acquisition. You can assign more than one subsequent process chain to a data transfer process.
Call monitor for data transfer process	*Monitor*		The system displays the monitor with the data transfer process requests for the current and previous day.
Trigger repair of incorrect DTP request	*Start Repair*	-	If the daemon ended the data transfer due to an error in the transformation, for example, and you have corrected the error, you can restart the data transfer with a repair. The repair deletes scheduled jobs for the daemon, deletes the DTP and change log requests, switches the DTP type to the standard, and repeats the data transfer for the requests in which errors occurred. You must then activate the data in the DataStore object manually.

DTP Request Functions

Function	Context Menu	Menu Path	Description/More Information
Call administration for DataStore object	*Manage InfoProviders*	-	-
Call monitor for DTP request	*Monitor*	-	The system displays the monitor for the selected request.
Close request	*Close Request*	-	You can close a request if the daemon has terminated. See Closing Requests in the Monitor for Real-Time Data Acquisition [Page 560].

Subsequent Process Chain Functions

Function	Context Menu	Menu Path	Description/More Information
Call process chain maintenance	*Maintain Subsequent Process Chain*	-	-

	Or double-click		
Delete assignment to daemon	*Delete Assignment*	-	The system no longer displays the process chain in the monitor for real-time data acquisition. The process chain does not start when the daemon closes a DTP request.

4.3.2.6 Monitoring Real-Time Data Acquisition

Use

You monitor the data transfer in the monitor for real-time data acquisition. Errors can occur when data is transferred to the PSA or when data is further updated from the PSA into the DataStore object.

If an error occurs, the daemon terminates the data transfer. You can analyze and correct the error, and then restart the data transfer.

Procedure

In the monitor for real-time data acquisition, the daemon displays the status.

If data transfer for a DataSource terminates and this is the only DataSource assigned to the daemon, the daemon terminates ( ). If other DataSources are assigned to the daemon, only the data transfer in which the error occurred terminates. The daemon remains active ( ) if no errors occur in the data transfer for at least one of the DataSources assigned to it.

Calling Logs for Daemons

1. Check the log for the daemon.

   In the context menu of the daemon, choose *Display Log*, or select the daemon and choose *Display Log.*

   The screen that appears displays all logs for the daemon.

   Using the application log (transaction SLG1), you can restrict the number of logs displayed to display those for a specific time period only. This makes the display easier to use.

   In the application log (transaction SLG1), choose **RSAP** as the object and **REALTIME** as the subobject. Enter the InfoPackage ID as the external identification and restrict the time period to the period for which you want to display logs.

2. Open the error log.
3. Use the messages provided to determine whether the error occurred during extraction or in the transformation.

Correcting Errors That Occur During Extraction

If the PSA request is red because errors occurred during extraction, for example the RFC connection was interrupted, the background job was deleted, or the system terminated the program, proceed as follows:

1. Check the data in the PSA.

   To do this, in the monitor for the PSA request, choose *PSA Maintenance*.

2. If the technical status of the request is red and you have resolved the error that occurred during extraction, proceed as follows: You delete the request from the PSA, repeat the delta update into the PSA, write the request to the DataStore object using the standard procedure, and activate the request in the DataStore object. You then set the data transfer back to real-time data acquisition and restart the daemon. The steps you take are as follows:

3. Delete the request from the PSA.

   To do this, in the monitor for the PSA request, choose *PSA Maintenance*. On the screen that appears, select the request containing errors and choose *Delete*.

4. To repeat the last delta update, in the Data Warehousing Workbench, create a standard InfoPackage for the DataSource, choose *Delta Update* on the *Update* tab page, and execute the InfoPackage.

5. Check the status of the PSA request in the extraction monitor.

6. If the data was successfully updated into the PSA, navigate to the maintenance for the assigned data transfer process and choose *Change to Standard DTP*. You can start the data transfer process on the *Execute* tab page.

7. Check the status of the DTP request in the administration for the DataStore object.

8. If the data was successfully updated into the DataStore object, select the request in DataStore object administration and activate the data by choosing *Activate*.

9. Navigate to data transfer process maintenance and choose *Change to Real-Time DTP*.

10. To reactivate the data transfer with real-time data acquisition, open the monitor for real-time data acquisition and start the daemon by choosing *Restart Batch Job*.

Resolving Errors that Occurred During Transformation

If the error occurred during the transformation, for example a routine in the transformation contains an error, the background job was deleted, or the system terminated the program, proceed as follows:

1. Check the status of the DTP request by choosing *Monitor* in the context menu of the request.

   If the status of the request is red, proceed as follows:

2. Check whether the data for the DTP request is correct in the DataStore object.

   To do this, in the monitor for the DTP request, choose *Administer Data Target*.

3. If the status of the request is red due to a technical problem in the transformation, for example, but the data in the DataStore object is correct, switch the status to green. Restart the daemon in the monitor for real-time data acquisition by choosing *Restart Batch Job* in the context menu.

4. If the data in the DataStore object contains errors, check the transformation and correct the errors.

5. In the monitor for real-time data acquisition, choose *Start Repair* in the context menu of the data transfer process.

   The repair process deletes all scheduled daemon jobs from the job list.

   The DTP request containing the errors and the change log request are both deleted.

   A standard DTP is created and executed. The data from the erroneous request is retransferred from the PSA into the DataStore object.

6. Check the status of the DTP request in DataStore object administration (by choosing *Manage InfoProvider* in the context menu of the data transfer process).

7. If the data was successfully updated into the DataStore object, select the request in DataStore object administration and activate the data by choosing *Activate*.

8. To reactivate the data transfer with real-time data acquisition, start the daemon in the monitor for real-time data acquisition by choosing *Restart Batch Job* in the context menu.

   The type for the data transfer process is then automatically set to *Real-Time Data Acquisition*.

   As an alternative to using the repair, you can manually repeat the data transfer process. To do this, you must manually execute the steps that the system executes during a repair.

4.3.2.7 Creating InfoPackages for Real-Time Data Acquisition

Use

You use (standard) InfoPackages to determine the parameters for data request and data transfer from a source to the inbound layer of BI, the Persistent Staging Area (PSA).

For real-time data acquisition (SAPI and Web service) and when using Web services for standard update (with a standard data transfer process), you use a special type of InfoPackage which allows you to specify the required parameters for these transfer types. This InfoPackage type is called *InfoPackage for Real-Time Data Acquisition*.

Requests that are instantiated using an InfoPackage for real-time data acquisition remain open across several load processes. In the InfoPackage for real-time data acquisition, you specify the size and time limits for requests and the maximum size of the data package. The system closes a request when one on the defined threshold values is reached.

For real-time data acquisition (SAPI and Web service), the daemon processes the InfoPackage.

Prerequisites

- The DataSource for which you want to create an InfoPackage supports real-time data acquisition.
- If you are transferring data into BI using the service API, you have already created an InfoPackage for initializing the delta process for this DataSource, and have completed the initialization.

Procedure

You can only create **one** InfoPackage for a DataSource for real-time data acquisition.

1. In the modeling view of the Data Warehousing Workbench, navigate to the DataSource tree of the required source system, select the DataSource, and in the context menu, choose *Create InfoPackage*.
2. In the following dialog box, enter a description for the InfoPackage. In the *DataSource* table, select the required DataSource.
   - If you are transferring the data using the service API:

     Select the *Real-Time Data Acquisition InfoPackage* indicator. This field is only input ready if the DataSource supports real-time data acquisition, an InfoPackage for real-time data acquisition has not yet been created for this DataSource, and the delta initialization is complete.
   - If you are transferring the data using the Web API:

     The *Real-Time Data Acquisition InfoPackage* indicated is already set.
3. Choose ✔ *Save*.

   The scheduler appears; you can now maintain InfoPackages.
4. Specify values for the parameters on the following tab pages:
   a. *Data Selection* tab page:
      - If you are transferring data using the service API, the selections from the delta initialization are displayed.
      - If you are transferring data using the Web service, you can specify the selection values for the data transfer.

        If selectable fields are defined in the Web service DataSource and you have specified corresponding selections in the InfoPackage, before writing to the PSA, the system filters out all data records that do not fulfill the selection conditions. In this case, the number of records available in the PSA is less than the number of records sent by the source.

   b. *Extraction* tab page:
      - If you are transferring the data using the service API, *Real-Time Extraction from SAP Source System* is displayed as the adapter.
      - If you are transferring the data using the Web API:

        Web Service (Push) is displayed as the adapter.

        Change the security profile for the data transfer, as required.

        If the name of the Web service or dependent objects needs to be predefined in the target system of a transport, enter a fixed name for the Web service. The Web service and any dependent objects, such as the corresponding function module, are then created using this name and the relevant generation prefix.

   c. *Processing* tab page:
      - Specify the maximum number of days or hours after which the system closes a request.

Irrespective of the number of records loaded, the system closes the request as soon as this threshold value is reached and the data transfer continues in a new request. The default setting is one day.

- Specify the maximum number of data records that is permitted for each request before a new request is opened.
- Specify how many failed attempts are permitted when the daemon is accessing the data before a request is given error status.
- The maximum size a data package for the data transfer can reach before a new package is opened is predefined. The default setting is 10,000 records. This setting can only be changed by the debug user. This setting may affect performance if you are updating data to the PSA using parallel processing (see Tasks on Demand [External] → Change System Parameters).

If a request is closed based on the settings you make, the system automatically starts a new request to continue the data transfer.

d. *Schedule* tab page:

To access the monitor for real-time data acquisition, choose *Assign*.

Result

In the monitor for real-time data acquisition, the system displays the InfoPackage in the *Unassigned Nodes* area, under the DataSource for which the InfoPackage was created. You can assign the InfoPackage to a daemon by choosing *Assign Daemon* in the context menu of the DataSource. The daemon can now use the InfoPackage for data extraction.

4.3.2.8 Creating Data Transfer Processes for Real-Time Data Acquisition

Use

You use the data transfer process (DTP) for real-time data acquisition to transfer data to the DataStore object from the PSA. In the DataStore object, the data is available for use in reporting.

Prerequisites

You have used transformations to define the data flow between the DataSource and the DataStore object.

The selections for the data transfer process do not overlap with selections in other data transfer processes.

Procedure

The starting point when creating a data transfer process is the DataStore object into which you want to transfer data. In the Data Warehousing Workbench, an object tree is displayed and you select the DataStore object.

1. In the context menu, choose *Create Data Transfer Process*.

The dialog box for creating a data transfer process appears.

2. Select *DTP for Real-Time Data Acquisition* as the *DTP Type*.
3. As the source object, select the DataSource from which you want to transfer data to the DataStore object.

   The input help for the source object shows the selection of DataSources that already exist in the data flow for the DataStore object. An additional *List* pushbutton is available. This allows you to select a DataSource from the complete list of BI DataSources.
4. Choose *Continue*.

   The data transfer process maintenance screen appears.

   The header data for the data transfer process shows the description, ID, version, and status of the data transfer process, along with the delta status.
5. On the *Extraction* tab page, specify the parameters:
   a. *Delta* is chosen as the extraction mode for real-time data acquisition.
   b. If necessary, determine filter criteria for the delta transfer. To do this, choose *Filter.*

      This means that you can use multiple data transfer processes with disjunctive selection conditions to efficiently transfer small sets of data from a source into one or more targets, instead of transferring large volumes of data. You can specify individual selections, multiple selections, intervals, selections based on variables, or routines. To change the list of InfoObjects that can be selected, choose *Change Selection*.

      The icon next to pushbutton *Filter* indicates that predefined selections exist for the data transfer process. The tool tip for this icon displays the selections as a character string.
   c. Semantic grouping is not used for real-time data acquisition. The data is read from the source in packages.
6. On the *Update* tab page, specify the parameters:

   Make the settings for error handling. You determine:

   - How you want to update valid records when errors occur.
   - How many errors can occur before the load process terminates.

     Note that this setting only has an impact while repairing a DTP request (repair) and during the conversion of the DTP to standard DTP (for example, to correct an error during extraction).

     For real-time data acquisition, specify in the InfoPackage the maximum number of failed attempts that is permitted when the daemon is accessing data before the data transfer terminates with an error.

     For more information about error handling settings, see Handling Data Records with Errors [Page 328].
7. On the *Execute* tab page, specify the parameters:

   On this tab page, the process flow of the program for the data transfer process is displayed in a tree structure.
   a. Specify the status that you want the system to adopt for the request if warnings are to be displayed in the log.

b. Specify how you want the system to determine the overall status of the request.

8. Check, save, and activate the data transfer process.

Result

To access the monitor for real-time data acquisition, choose *Real-Time Monitor*. In the context menu of a DataSource, you can assign the data transfer process to a daemon. The daemon uses the data transfer process to process data.

4.3.3 Open Hub Destination

Definition

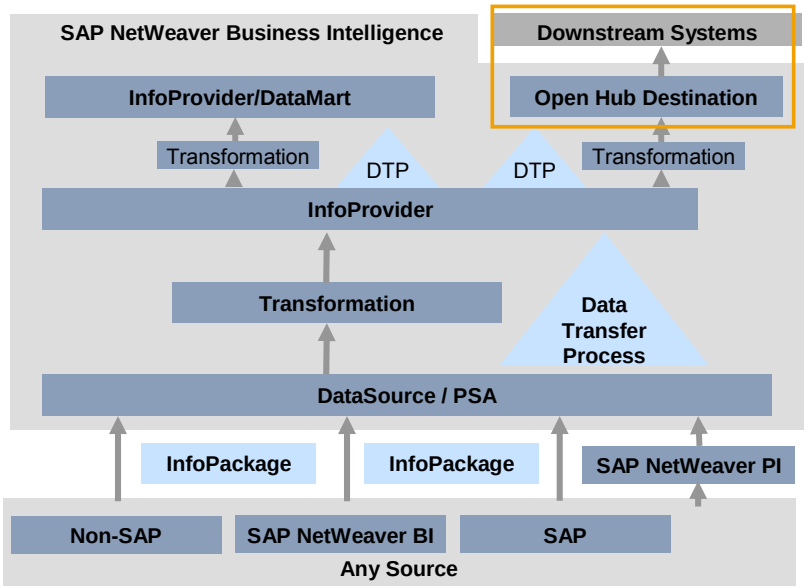
The open hub destination is the object that allows you to distribute data from a BI system to non-SAP data marts, analytical applications, and other applications. It allows you to ensure controlled distribution across multiple systems.

The open hub destination defines the target to which the data is relayed.

In earlier releases, the open hub destination was part of the InfoSpoke. It is now an independent object that provides more options as a result of its integration into the data flow.

The open hub service previously provided with the InfoSpoke can still be used. We recommend, however, that you use the new technology to define new objects.

The following figure outlines how the open hub destination is integrated into the data flow:



Use

Database tables (in the database for the BI system) and flat files can act as open hub destinations. You can extract the data from a database to non-SAP systems using APIs and a third-party tool.

Structure

The open hub destination contains all the information about a data target: the type of destination, the name of the flat file or database table and their properties, and the field lists and its properties.

Integration

You can use the data transfer process to update data to the open hub destination. This involves a transformation of the data. Not all rules types are available in the transformation for an open hub destination: Reading master data, time conversion, currency translation and unit conversion are not available.

4.3.3.1 Creating Open Hub Destinations

Procedure

1. In the *Modeling* area of the Data Warehousing Workbench, choose the open hub destination tree.
2. In the context menu of your InfoArea, choose *Create Open Hub Destination*.
3. Enter a technical name and a description. We recommend that you use the object from which you want to update data to the open hub destination as the template.
4. On the *Destination* tab page, select the required destination. The other settings you can make on this tab page differ depending on the destination you select. For more information, see
   - Database Tables As Destinations [Page 653]
   - Files As Destinations [Page 653]
   - Third-Party Tools As Destinations [Page 654]
5. On the *Field Definition* tab page, edit the field list. For more information, see Field Definitions [Page 656].
6. Activate the open hub destination.

Result

You can now use the open hub destination as a target in a data transfer process.

4.3.3.2 Database Tables As Destinations

Use

You can select a database table as an open hub destination.

Features

Generating Database Tables

When you activate the open hub destination, the system generates a database table. The generated database table has the prefix /BIC/OHxxx (xxx is the technical name of the destination).

Deleting Data from the Table

With an extraction to a database table, you can either retain the history of the data or just store the new data in the table. Choose *Delete Data from Table* when defining your destination if you want to overwrite the fields. In this case, the table is completely deleted and regenerated before each extraction takes place. We recommend that you use this mode if you do not want to store the history of the data in the table. If you do not select this option, the system only generates the table once before the first extraction. We recommend that you use this mode if you want to retain the history of the extracted data.

Note that if changes are made to the properties of the database table (for example, fields are added), the table is always deleted and regenerated.

Table Key Fields

You can choose whether you want to use a technical key or a semantic key.

Technical key:

If you set the *Technical Key* indicator, a unique key is added that consists of the technical fields OHREQUID (open hub request SID), DATAPAKID (data package ID), and RECORD (sequential number of a data record to be added to the table within a data package). These fields display the individual key fields for the table.

Using a technical key with a target table is particularly useful if you want to extract into a table that is not deleted before extraction. If an extracted record has the same key as a record that already exists, the duplicate records cause a short dump.

Semantic key:

If you set the *Semantic Key* indicator, the system selects all the fields in the field list as semantic keys, if they are suitable. You can change this selection in the field list. However, note that duplicate records may result from using a semantic key.

4.3.3.3 Files As Destinations

Use

You can select flat files in format *.CSV* as an open hub destination.

Features

The only file format that is supported for extraction to flat files is *.CSV*. A control file with information about the metadata is also generated. You can either save the file on the application server or in a local directory.

If you save the file locally, the file size must not exceed a half GB. When transferring mass data, you should save the file on the application server.

If the data is to be written to a BI system application server, you can determine the file name in two ways:

- File name:

  The file name is made up of the technical name of the open hub destination and the suffix *.CSV*. You cannot change this name.

- Logical file name:

  You can use input help to select a logical file name that you have already defined in Customizing. Create a logical path and assign a logical file name to it (see Defining Logical Path and File Names [External]).

  A logical file name can be made up of fixed path information, but also of variables such as calendar day and time. Logical file names can be transported.

If you save the file in a local directory, you cannot change the name of the file. It is made up of the technical name of the open hub destination and the suffix *.CSV*. The associated control file also has the prefix S_.

4.3.3.4 Third-Party Tools As Destinations

Use

You can use the open hub destination to extract data to non-SAP systems. Various APIs allow you to connect a third-party tool to the BI system and to use this third-party tool to distribute data to other non-SAP systems.

Features

First you extract the data from BI InfoProviders or DataSources into a database table in the BI system. The third-party tool receives a message when the extraction process is complete. You can define parameters for the third-party tool. You can also use the monitor to oversee the process.

You can connect one or more data transfer processes to an open hub destination of type *Third-Party Tool*.

You can use a process chain to start the extraction process not only in the BI system itself, but also using the third-party tool.

The following APIs are available:

RSB_API_OHS_DEST_SETPARAMS: You use this API to transfer the parameters of the third-party tool that are required for the extraction to the BI system. These parameters are saved in a parameter table within the BI system in the metadata for the open hub destination.

RSB_API_OHS_3RDPARTY_NOTIFY: This API sends a message to the third-party tool after extraction. It transfers the open hub destination, the request ID, the name of the database

table, the number of extracted data records and the time stamp. In addition, you can add another parameter table that contains the parameters that are only relevant for the third-party tool.

RSB_API_OHS_REQUEST_SETSTATUS: This API sets the status of extraction to the third-party tool in the monitor. Red means that the existing table is not overwritten. If the status is green, the request is processed further. Normally the user can change the status manually in the monitor or in the maintenance screen for the data transfer process. However, these manual functions are deactivated with open hub destinations of type *Third-Party Tool*.

RSB_API_OHS_DEST_GETLIST: This API delivers a list of all open hub destinations.

RSB_API_OHS_DEST_GETDETAIL: This API gets the details of an open hub destination.

RSB_API_OHS_DEST_READ_DATA: This API reads data from the database table in the BI system.

For information on the parameters of the APIs, see:

API: RSB_API_OHS_DEST_SETPARAMS [External]

API: RSB_API_OHS_3RDPARTY_NOTIFY [External]

API: RSB_API_OHS_REQUEST_SETSTATUS [External]

API: RSB_API_OHS_DEST_GETLIST [External]

API: RSB_API_OHS_DEST_GETDETAIL [External]

API: RSB_API_OHS_DEST_READ_DATA [External]

Process Flow:

Extraction to the third-party tool can be executed as follows:

1. You define an open hub destination with *Third-Party Tool* as the destination type.
2. You create an RFC destination for your third-party tool and enter it in the definition of the open hub destination.
3. You use API RSB_API_OHS_DEST_SETPARAMS to define the parameters for the third-party tool that are required for the extraction.
4. You either start extraction immediately or include it in a process chain. You can also start this process chain from the third-party tool using process chain API RSPC_API_CHAIN_START. The extraction process then writes the data to a database table in the BI system.
5. When the extraction process is finished, the system sends a notification to the third-party tool using API RSB_API_OHS_3RDPARTY_NOTIFY.
6. The extracted data is read by API RSB_API_OHS_DEST_READ_DATA.
7. The status of the extraction is transferred to the monitor by API RSB_API_OHS_REQUEST_SETSTATUS.

See also:

For detailed information about certification and the scenario, see the SDN at **`www.sdn.sap.com`** → *Partners and ISVs* → *SAP Integration and Certification Center* → *Integration Scenarios* → *Business Intelligence* → *Interface: BW-OHS*.

4.3.3.5 Field Definition

Use

On the *Field Definition* tab page you define the properties of the fields that you want to transfer.

Features

We recommend that you use a template as a basis when you create the open hub destination. The template should be the object from which you want to update the data. This ensures that all the fields of the template are available as fields for the open hub destination. You can edit the field list by removing or adding fields. You can also change the properties of these fields.

You have the following options for adding new fields:

- You enter field names and field properties, independent of a template.
- You select an InfoObject from the *Template InfoObject* column. The properties of the InfoObject are transferred into the rows.
- You choose *Select Template Fields*. A list of fields are available as fields for the open hub destination that are not contained in the current field list. You transfer a field to the field list by double-clicking on it. This allows you to transfer fields that had been deleted back into the field list.

If you want to define the properties of a field so that they are different from the properties of the template InfoObject, delete the template InfoObject entries for the corresponding field and change the properties of the field. If there is a reference to a template InfoObject, the field properties are always transferred from this InfoObject.

The file or database table that is generated from the open hub destination is made up of the fields and their properties and not the template InfoObjects of the fields.

If the template for the open hub destination is a DataSource, field SOURSYSTEM is automatically added to the field list with reference to InfoObject 0SOURSYSTEM. This field is required if data from heterogeneous source systems is being written to the same database table. The data transfer process inserts the source system ID that is relevant for the connected DataSource. You can delete this field if it is not needed.

If you have selected *Database Table* as the destination and *Semantic Key* as the property, the field list gets an additional column in which you can define the key fields for the semantic key.

In the *Format* column, you can specify whether you want to transfer the data in the internal or external format. For example, if you choose *External Format* here, leading zeros will be removed from a field that has an ALPHA conversion routine when the data is written to the file or database table.

4.3.4 ICF Services in BI

Use

Services are delivered with SAP NetWeaver Business Intelligence that are based on the Internet Communication Framework [External] (ICF) in the SAP NetWeaver Application

Server (Web AS). ICF services are HTTP services as they serve to execute HTTP request handlers.

The BI HTTP services allow you to display or exchange BI data using a URL. Some of these services are implemented as Web services.

Integration

The URL of an HTTP service delivered in a namespace of BI is structured according to the following schema:

<Protocol>://<Server>:<Port>/sap/bw/<Service>

URL Prefix

The values used for the place holder in the specified URL schema depend on the particular installation. As **`<Protocol>`** *http* and *https* can be selected. Enter your message server as **`<Server>`**.

You can check which URL prefix your BI system generated as follows:

1. Call the *Function Builder* (transaction code SE37).
2. Enter the name of the function module `RSBB_URL_PREFIX_GET`.
3. Choose *Test/Execute.* The *Test Function Module* screen appears.
4. As import parameter I_HANDLERCLASS, specify the name of the ICF handler (HTTP-Request Handler) for the required service.

   You can find out the name of the ICF handler in *Maintenance of Services* (transaction code SICF): In the HTTP service tree, navigate to the required service component. Double click to open the dialog box *Change/Create a Service*. The HTTP request handler for the service is displayed on the *Handler List* tab page.
5. Choose *Execute*. The export parameter `E_URL_PREFIX` contains the generated URL prefix.

Service

Specify the technical name of the required service here. The name comprises all the elements of the path in the HTTP service tree (transaction code SICF).

Prerequisites

The required HTTP service must be active.

To check this, in *Maintenance of Services* (transaction code SICF), navigate to the required service component. If the service is active, you **cannot** select the *Activate Service* entry in the context menu.

Features

BI provides the following HTTP services:

- Display of metadata for BI objects (see Calling the Metadata Repository as an HTTP Service [External])

- Exchange of metadata between different systems (see Exchanging Metadata in XML Format [External])
- Display of user-defined online documentation on metadata objects (see Document as Online Documentation [External])

The following services are implemented as Web services:

- Open Analysis Interfaces (see XML for Analysis [External])

See also:

The two services for the status and tracking system of BW-BPS are not located in the BI namespace. Instead they can be found at **`sap/bc/bsp/sap/tunguska`** and **`tunguska_detail`**. For more information, see Maintaining Web Services for STS [External].

The Web services that you created are also not located in the BI namespace. For more information on these Web services, see Transfer of Data Using Web Service [Page 571].

4.4 Considering Performance

Purpose

Note the following points to ensure optimal system performance when you develop:

Modeling BI Objects

To optimize database performance for BI objects, you can set various DB storage parameters for each object.

For more information, see DB Storage Parameters [Page 659].

Depending on the database, you can improve access to data by using partitioning or clustering.

For more information, see Partitioning [Page 660] and Clustering [Page 662].

For DataStore objects, you can apply various object-specific settings.

For more information, see Performance Optimization for DataStore Objects [Page 663].

You can improve the performance of InfoCubes by creating aggregates or BI accelerator indexes.

For more information, see Performance Optimization with Aggregates [Page 665] and Performance Optimization with a BI Accelerator [Page 700].

You can also apply various object-specific settings.

For more information, see Compressing InfoCubes [Page 735] and InfoCube Performance [Page 736].

Modeling Non-Cumulatives

To attain good performance for a query based on InfoCubes with non-cumulative key figures, there are various factors to consider.

For more information, see Performance Tips [Page 434].

Archiving

Data archiving and storage allows you to simplify InfoCube and DataStore object administration and improve system performance.

For more information, see Information Lifecycle Management [Page 738].

Enhancements Using Routines

The general performance guidelines for ABAP development apply when you develop routines. For more information, see ABAP Programming (BC-ABA) [External] and Performance Notes [External].

Customer Exits and BAdIs

The general performance guidelines for ABAP development apply when you implement customer exits. For more information, see ABAP Programming (BC-ABA) [External] and Performance Notes [External].

You use ABAP objects to implement BAdIs. For more information, see ABAP Objects [External].

ICF Services

The general guidelines for ICF services also apply when you develop ICF services in the BI system. For more information, see Development [External].

4.4.1 DB Storage Parameters

Use

You can maintain database storage parameters for PSA tables, master data tables, InfoCube fact- and dimension tables, and DataStore object tables.

Use this setting to determine how the system handles the table when it creates it in the database:

- Use *Data Type* to set in which physical database area (tablespace) the system is to create the table. Each data type (master data, transaction data, organization- and Customizing data, customer data) has its own physical database area, in which all tables assigned to this data type are stored. If selected correctly, your table is automatically assigned to the correct area when it is created in the database.

  We recommend you use separate tablespaces for very large tables.

  You can find information about creating a new data type in SAP Note 0046272 (Introduce new data type in technical settings).

- Via *Size Category*, you can set the amount of space the table is thought to need in the database. Five categories are available in the input help. You can also see here how many data records correspond to each individual category. When creating the table, the system reserves an initial storage space in the database. If the table later requires more storage space, it obtains it as set out in the size category. Correctly

setting the size category prevents there being too many small extents (save areas) for a table. It also prevents the wastage of storage space when creating extents that are too large.

You can use the maintenance for storage parameters to better manage databases that support this concept.

You can find additional information about the data type and size category parameters in the ABAP Dictionary table documentation, under Technical Settings [External].

PSA Table

For PSA tables, you access the database storage parameter maintenance by choosing *Goto → Technical Attributes* in DataSource maintenance. In dataflow 3.x, you access this setting in transfer rule maintenance in the *Extras* menu.

You can also assign storage parameters for a PSA table already in the system. However, this has no effect on the existing table. If the system generates a new PSA version (a new PSA table) due to changes to the DataSource, this is created in the data area for the current storage parameters.

InfoObject Tables

For InfoObject tables, the maintenance of database storage parameters can be found in the InfoObject maintenance, in the *Extras* menu.

InfoCube/aggregate fact and dimension tables

For fact- and dimension tables, the maintenance of database storage parameters can be found in the InfoCube maintenance, in the *Extras* menu.

DataStore Object Tables (Activation Queue and Table for Active Data)

For DataStore object tables, the maintenance of database storage parameters can be found in DataStore object maintenance, in the *Extras* menu.

4.4.2 Partitioning

Use

You use partitioning to split the total dataset for an InfoProvider into several, smaller, physically independent and redundancy-free units. This separation improves system performance when you analyze data delete data from the InfoProvider.

Integration

All database providers except DB2 for Linux, UNIX, and Windows support partitioning. You can use clustering to improve the performance for DB2 for Linux, UNIX, and Windows.

If you are using IBM DB2 for i5/OS as the DB platform, you require database version V5R3M0 or higher and an installation of component DB2 Multi System. Note that with this system constellation the BI system with active partitioning can only be copied to other IBM iSeries with an SAVLIB/RSTLIB operation (homogeneous system copy). If you are using this database you can also partition PSA tables. You first have to activate this function using

RSADMIN parameter DB4_PSA_PARTITIONING = 'X'. SAP Note 815186 includes more comprehensive information on this.

Prerequisites

You can only partition a dataset using one of the two partitioning criteria ‘calendar month’ (0CALMONTH) or ‘fiscal year/period (0FISCPER). At least one of the two InfoObjects must be contained in the InfoProvider.

If you want to partition an InfoCube using the fiscal year/period (0FISCPER) characteristic, you have to set the fiscal year variant characteristic to constant.

See Partitioning InfoCubes using Characteristic 0FISCPER [Page 140].

Features

When you activate the InfoProvider, the system creates the table on the database with one of the number of partitions corresponding to the value range. You can set the value range yourself.

Choose the partitioning criterion 0CALMONTH and determine the value range

From 01.1998

To 12.2003

6 years x 12 months + 2 = 74 partitions are created (2 partitions for values that lay outside of the range, meaning < 01.1998 or >12.2003).

You can also determine the maximum number of partitions created on the database for this table.

Choose the partitioning criterion 0CALMONTH and determine the value range

From 01.1998

To 12.2003

Choose 30 as the maximum number of partitions.

Resulting from the value range: 6 years x 12 calendar months + 2 marginal partitions (up to 01.1998, from 12.2003) = 74 single values.

The system groups three months together at a time in a partition (meaning that a partition corresponds to exactly one quarter); in this way, 6 years x 4 partitions/year + 2 marginal partitions = 26 partitions created on the database.

The performance gain is only achieved for the partitioned InfoProvider if the time characteristics of the InfoProvider are consistent. This means that with a partition using 0CALMONTH, all values of the 0CAL x characteristics of a data record have to match.

In the following example, only record 1 is consistent. Records 2 and 3 are not consistent:

0CALYEAR	0CALMONTH	0CALDAY
1998	05.1998	28.05.1998
1998	00.0000	00.00.0000
1998	05.1998	01.01.1998

Note that you can only change the value range when the InfoProvider does not contain data. If data has already been loaded to the InfoProvider, you have to perform repartitioning.

For more information, see Repartitioning [Page 141].

We recommend that you use “partition on demand“. This means that you should not create partitions that are too large or too small. If you choose a time period that is too small, the partitions are too large. If you choose a time period that ranges too far into the future, the number of partitions is too great. Therefore we recommend that you create a partition for a year, for example, and that you repartition the InfoProvider after this time.

Activities

In InfoProvider maintenance, choose *Extras* → *Partitioning* and specify the value range. Where necessary, limit the maximum number of partitions.

4.4.3 Clustering

Use

Clustering allows you to save sorted data records in the fact table of an InfoCube. Data records with the same dimension keys are saved in the same extents (related database storage unit). This means that same data records are not spread across a large memory area and thereby reduces the number of extents that the system has to read when it accesses tables. This greatly accelerates read, write and delete access to the fact table.

Prerequisites

Currently the function is only supported by the database platform DB2 for Linux, UNIX, and Windows. You can use partitioning to improve the performance of other databases. For more information, see Partitioning [Page 660].

Features

Two types of clustering are available: Index clustering and multidimensional clustering (MDC).

Index Clustering

Index clustering organizes the data records of a fact table according to the sort sequence of an index. Organization is linear and corresponds to the values of the index field.

If a data record cannot be inserted in accordance with the sort sequence because the relevant extent is already full, the data record is inserted into an empty extent at the end of

the table. For this reason, the system cannot guarantee that the sort sequence is always correct, particularly if you perform many insert and delete operations. Reorganizing the table restores the sort sequence and frees up memory space that is no longer required.

The clustering index of an F fact table is, by default, the secondary index in the time dimension. The clustering index of an E fact table is, by default, the acting primary index (P index).

As of release SAP BW 2.0, index clustering is standard for all InfoCubes and aggregates.

Multidimensional Clustering (MDC)

Multidimensional clustering organizes the data records of a fact table in accordance with one or more fields that you define freely. The selected fields are also marked as MDC dimensions. Only data records that have the same values in the MDC dimensions are saved in an extent. In the context of MDC, an extent is called a block. The system can always guarantee that the sort sequence is correct. Reorganizing the table is not necessary, even with many insert and delete operations.

Block indexes from within the database, instead of the default secondary indexes, are created for the selected fields. Block indexes link to extents instead of data record numbers and are therefore much smaller. They save memory space and the system can search through them more quickly. This accelerates table requests that are restricted to these fields.

You can select the key fields of the time dimension or any customer-defined dimensions of an InfoCube as an MDC dimension. You cannot select the key field of the package dimension; it is automatically added to the MDC dimensions in the F fact table.

You can also select a time characteristic instead of the time dimension. In this case, the fact table has an extra field. This contains the SID values of the time characteristic. Currently only the time characteristics *Calendar Month* (0CALMONTH) and *Fiscal Year/Period* (0FISCPER) are supported. The time characteristic must be contained in the InfoCube. If you select the *Fiscal Year/Period* (0FISCPER) characteristic, a constant must be set for the *Fiscal Year Variant* (0FISCVARNT) characteristic.

Clustering is applied to all the aggregates of the InfoCube. If an aggregate does not contain an MDC dimension of the InfoCube, or if all the InfoObjects of an MDC dimension are created as line item dimensions in the aggregate, the aggregates are clustered using the remaining MDC dimensions. Index clustering is used for the aggregate if the aggregate does not contain any MDC dimensions of the InfoCube, or if it only contains MDC dimensions.

Multidimensional clustering was introduced in Release SAP NetWeaver 2004s and can be set up separately for each InfoCube.

For procedures, see Definition of Clustering [Page 144].

4.4.4 Performance Optimization for DataStore Objects

To ensure good performance when loading data into DataStore objects, we recommend that you note the following:

1. **Creating SID values**

   It takes a long time to generate SID values and you can avoid this in the following cases:

   - The *Generation of SID Values* indicator should not be set if you are using the DataStore object for data storage purposes only. If you do set this indicator, SIDs are created for all new characteristic values.

- If you are using line items (for example, document number or time stamp) as characteristics in the DataStore object, in characteristic maintenance, set the indicator to show that they are *Attribute Only*.

SID values are created in parallel if parallel activation is switched on (see last point). They are generated in the same number of parallel processes as are used for activation.

2. **Clustering and partitioning in active data tables (A tables)**

   You can speed up the deletion of data from the DataStore object and access to it for the data analysis by clustering and partitioning on the database. As the clustering or partitioning criterium, choose the characteristic that you want to use as the reference for deletion. For more information see Multi-Dimensional Clustering [Page 161] and Partitioning [Page 660].

3. **Indexing**

   Use selection criteria for queries that are based on DataStore objects. If key fields are specified, the existing primary index is used. The more frequently accessed characteristic should appear on the left-hand side.

   If you did not specify the key fields completely in the selection criteria (you can check this in the SQL trace), you can improve the runtime of the query by creating additional indexes. You create these secondary indexes in DataStore object maintenance.

   However, you should note that load performance is also affected if you have too many secondary indexes.

4. **Runtime parameters of DataStore objects**

   To improve performance when processing data in the DataStore object, you can make a number of settings for each DataStore object in the runtime parameters or maintain general default values for DataStore objects. You navigate to runtime parameter maintenance either in Customizing by choosing *SAP Customizing Implementation Guide → SAP NetWeaver → Business Intelligence → Performance Settings → Maintain Runtime Parameters of DataStore Objects* or in the *Administration* area of the Data Warehousing Workbench by choosing *Current Settings → DataStore Objects.*

   You can define the following:

   Parameters for activation:

   - The minimum number of data records for each data package when the data in the DataStore object is activated. In this way, you define the size of the data packages that are activated.
   - The maximum wait time in seconds when data in the DataStore object is activated. This is the time that the main process (batch process) waits for the split process before it determines that it has failed.

   Parameters for SID generation:

   - Minimum number of data records for each data package when SIDs are generated in the DataStore object.
   - Maximum wait time in seconds when SIDs are generated in the DataStore object.

   Parameters for rollback:

   - Maximum wait time in seconds for rollback in the DataStore object.

   Process parameters:

   - Type of processing: serial, parallel (batch), parallel (dialog)

- Maximum number of parallel processes
- Server group to be used for parallel processing of data in DataStore objects. You have to create the server groups first using the following path: *SAP Easy Access* → *Tools* → *Administration* → *Network* → *RFC Destination, RFC* → *RFC Groups*. If you do not specify anything here, processing runs on the server on which the batch activation process was started. If a server in the server group is not active, processing is terminated.

4.4.5 Performance Optimization with Aggregates

Purpose

Relational aggregates allow you to improve the performance of BI queries when data is read from an InfoCube. The data of a BI InfoCube is saved in relational aggregates in an aggregated form. Relational aggregates are useful if you want to improve the performance of one or more specific BI queries, or specifically improve the performance of reporting on characteristic hierarchies.

In all other cases, if relational aggregates are not sufficient, are too complex, or have other disadvantages, we recommend that you use BI accelerator (see Performance Optimization with BI Accelerator [Page 700]).

Integration

Accessing the OLAP Processor

You can use relational aggregates and a BI accelerator index for the same InfoCube. A BI query always tries to use performance-optimized sources by checking the sources from which it can draw the requested data. It checks the sources in the following order:

1. OLAP cache
2. BI accelerator index
3. Relational aggregates from the database
4. InfoCubes from the database

If an active BI accelerator index exists, the OLAP processor always accesses this BI accelerator index and not the relational aggregates. Therefore, with regard to modeling, we recommend that you create either relational aggregates or a BI accelerator index for an InfoCube.

Query Execution

When the query is executed, it is clear to the user whether data is being read from an aggregate, a BI accelerator index, or an InfoCube.

In the maintenance transaction, you can deactivate one or more aggregates on a temporary basis to test them for performance purposes or to analyze data consistency.

You can also execute the relevant query in the **query monitor** (transaction RSRT) using a corresponding debug option: Choose *Execute + Debug*. In the *Debug Options* dialog box, choose *Do Not Use Aggregates* to execute the query with an InfoCube, as long as no BI accelerator index exists.

4.4.5.1 Aggregates

Definition

An aggregate is a materialized, aggregated view of the data in an InfoCube. In an aggregate, the dataset of an InfoCube is saved redundantly and persistently in a consolidated form on the database.

Use

Aggregates allow quick access to InfoCube data during reporting. Similar to database indexes, they serve to improve performance.

Aggregates are particularly useful in the following cases:

- Executing and navigating in query data leads to delays if you have a group of queries
- You want to speed up the execution and navigation of a specific query
- You often use attributes in queries
- You want to speed up reporting with characteristic hierarchies by aggregating specific hierarchy levels

Structure

An aggregate is made up of the characteristics and navigation attributes that belong to an InfoCube. Characteristics that are **not** used in the aggregate are compressed.

Each component of an aggregate has to be assigned to a **selection type**. A selection type indicates the degree of detail to which the data in the underlying InfoCube is aggregated. You can choose one of the following selection types:

- All characteristic values ("*"): The data is grouped by all values of the characteristic or navigation attributes (see Selection Type "All Characteristic Values" (‘*’) [Page 667]).
- Hierarchy level (“H”): The data is grouped by the hierarchy level node. You can also store values on the hierarchy levels of an external hierarchy (see Selection Type "Hierarchy Level" (‘H’) [Page 669]).
- Fixed value (‘F’): The data is filtered by a single value (see Selection Type "Fixed Value" (‘F’) [Page 669]).

You can use both time-dependent attributes and time-dependent hierarchies in aggregates.

Integration

Access to Aggregates During Reporting

If you have created an aggregate for an InfoCube, activated it and entered data for it, the OLAP processor can access these aggregates automatically. For more information about the order in which the OLAP processor accesses the aggregates, see Performance Optimization with Aggregates [Page 665]. The different results are consistent when you navigate in the BI query. The aggregate is transparent for the user.

System Response Upon Changes to Data

New data is loaded to an aggregate at a defined time using logical data packages (requests). After this operation (roll up), the new data is available in reporting (see System Response Upon Changes to Master Data and Hierarchies [Page 689])

See also:

Performance Tuning for Queries with BI Aggregates in SAP Service Marketplace on the SAP NetWeaver Business Intelligence performance page.

4.4.5.1.1 Selection Type "All Characteristic Values" ('*')

The following examples are based on a simplified InfoCube (sales figures), containing the characteristics COUNTRY and CUSTOMER and the key figure SALES.

Example InfoCube

COUNTRY	CUSTOMER	SALES
USA	Buggy Soft	10
Germany	Ocean Networks	15
USA	Funny Duds	5
Austria	Ocean Networks	10
Austria	Thor Industries	10
Germany	Funny Duds	20
USA	Buggy Soft	25

Example 1: Aggregate Country ''*

- Characteristic COUNTRY is used for grouping
- Compressed using characteristic CUSTOMER

COUNTRY	SALES
USA	40
Germany	35
Austria	20

The data for key figure SALES is listed for the sum of the sales for each country and not for individual customers.

The aggregate can be used

- In a query that determines the sales for each country or the total sales
- For evaluations based on a navigation attribute for characteristic COUNTRY or a hierarchy of the countries

You cannot use the aggregate if characteristic CUSTOMER is used for drilldown or is selected in a query because the aggregate does not contain any information on the customer.

Example 2: Aggregate Industry ''*

In the second example for the selection type *All Characteristic Values*, INDUSTRY is required as the **navigation attribute**. We use the following master data table for this purpose:

Master data table: customer

CUSTOMER	INDUSTRY
Buggy Soft	Technology
Ocean Networks	Technology
Funny Duds	Consumer products
Thor Industries	Chemical industry

- Grouped using navigation attribute INDUSTRY
- Compressed using characteristic CUSTOMER

Industry aggregate

INDUSTRY	SALES
Technology	60
Consumer products	25
Chemical industry	10

The aggregate can be used

- In a query that is used to determine the sales for each industry or the total sales
- For evaluations based on node values, if a hierarchy exists for the industries

With the exception of the industry which is assigned to each respective customer, you cannot retrieve detailed information for characteristic CUSTOMER.

Example 3: Aggregate Sales Person ''*

In the third example for the selection type 'All Characteristic Values', the **time-dependent navigation attribute** SALESPERSON is required. We use the following table for this purpose:

Sales Person Table

COUNTRY	FROM	TO	SALES PERSON
USA	01.01.2000	31.12.2000	Smith
USA	01.01.2000	31.12.2001	Miller
Germany	01.01.2000	31.03.2001	Meyer
Germany	01.04.2000	31.12.2001	Huber
Austria	01.01.2000	31.12.2001	Huber

- Navigation attribute SALES PERSON is used for grouping
- Characteristic COUNTRY is used for compressing

- Key date: Sept. 01, 2001

SALES PERSON	SALES
Miller	40
Huber	55

The aggregate can be used in a query that has the same key date as the aggregate.

4.4.5.1.2 Selection Type "Fixed Value" (‘F’)

The following example is based on a simplified InfoCube (sales figures) containing characteristics COUNTRY and CUSTOMER as well as key figure SALES (see Selection Type "All Characteristic Values" (‘*’) [Page 667]),

Example: Aggregate Country 'F' = DE; Customer ''*

The aggregate only contains data for characteristic value 'Germany'.

Fixed Value: Germany

COUNTRY	CUSTOMER	SALES
Germany	Ocean Networks	15
Germany	Funny Duds	20

The aggregate can only be used in queries that have the same filter value.

The aggregate cannot be used if other countries are required in a query.

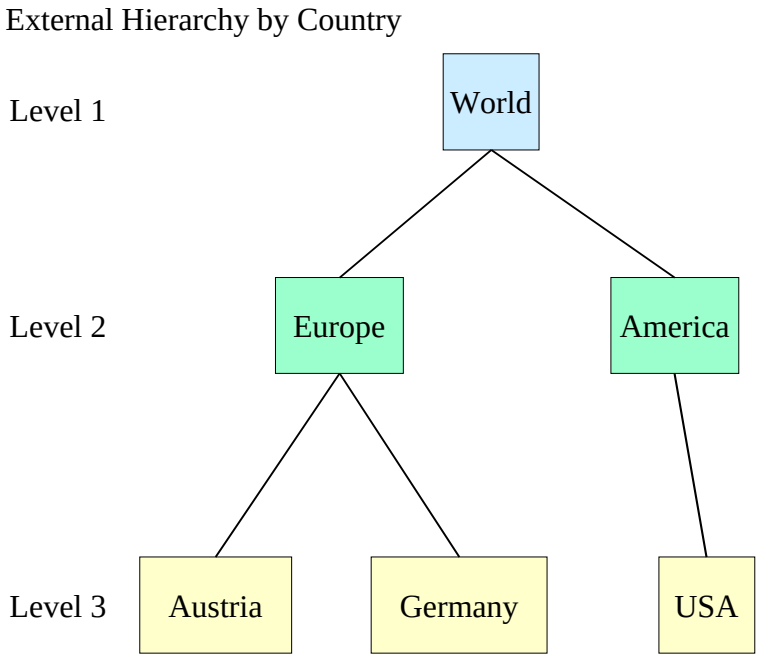
We recommend that you use filter values for aggregates if only one variant is needed for reporting, for example:

- The plan/actual indicator
- The current fiscal year
- Specific versions

4.4.5.1.3 Selection Type "Hierarchy Level" ('H').

The following example is based on:

- The simplified InfoCube (sales figures) containing characteristics COUNTRY and CUSTOMER as well as key figure SALES (see Selection Type "All Characteristic Values" (‘*’) [Page 667])
- The following external hierarchy for the countries:



Example: Aggregate Country 'H' Level 2

- Characteristic COUNTRY is used for compression on level 2 of the country hierarchy (Europe, America)
- The characteristic CUSTOMER is not contained in the aggregate

Hierarchy level 2

COUNTRY	SALES
America	40
Europe	55

This aggregate can be used in queries

- That require sales for a node of the hierarchy on level 2 or higher, including queries that have the country hierarchy as presentation hierarchy in the drilldown and are not expanded deeper than level 2
- That determine the total sum of the sales if:
  - The hierarchy does not have multiple leaves, in other words, each variant of a characteristic occurs only once in the hierarchy
  - Node "non-assigned" is not suppressed

The aggregate cannot be used if the third level of the hierarchy (Austria, Germany, USA) is required in a query.

4.4.5.2 Create and Change

Use

The following sections explain how you create aggregates and edit them.

- If you have not yet created aggregates for an InfoCube but you want to optimize query performance for it, see Creating the First Aggregate for an InfoCube [Page 671].
- For more information about manually creating and changing aggregates, see Editing Aggregates Manually [Page 672].
- On the *Maintenance of Aggregates* screen, you can find detailed information about the structure and status of the individual aggregates. See Design and Components of Aggregates [Page 676].
- Tools are available as further editing functions for aggregates [Page 679] to temporarily deactivate or completely delete aggregates that you no longer want to use in reporting.
- If you want the system to propose aggregates, see Automatic Selection and Optimization of Aggregates [Page 680].

4.4.5.2.1 Creating the First Aggregate for an InfoCube

Prerequisites

The InfoCube for which you want to create an aggregate has been saved and is active. There are not yet any aggregates for the selected InfoCube.

Note that the following prerequisites have to be filled if you want the system to propose aggregates:

- You must have created at least one query for the selected InfoCube or a MultiCube that uses the InfoCube. The system can propose the required aggregates when you start the queries and navigate in them.
- Make sure that you have switched on BI statistics collection for the selected InfoCube in the OLAP area. For more information, see Activating Data Transfer for BI Statistics [External].

Procedure

1. In the context menu for the InfoCube, choose *Maintain Aggregate*. The *Proposals for Aggregates* dialog box appears.
2. Specify if you want the system to propose aggregates.

Function	What You Need to Know
Create proposals	The system proposes suitable aggregates. The *Specify Statistics Data Evaluation* dialog box appears. Enter the required data. Choose *Continue.* The *Maintain Aggregates* screen appears. The

	system displays the proposed aggregates in the right area of the *Aggregates* screen. You can change these proposals by using Drag & Drop to add or remove dimensions, characteristics or attributes.
Create yourself	The *Maintain Aggregates* screen appears. For more information see Editing Aggregates Manually [Page 672].

3. To check the aggregate definition for inconsistencies, choose *Check Definition*.
4. Save the aggregates.

Result

You can activate the aggregates you have created and fill them with data.

4.4.5.2.2 Editing Aggregates Manually

Prerequisites

The InfoCube for which you want to create an aggregate has been saved and is active.

Procedure

How you manually create or change an aggregate for an InfoCube is described below.

Access from Data Warehousing Workbench

1. You are in the Data Warehousing Workbench in the *Modeling* functional area. In the navigation window, choose *InfoProvider* and in the InfoProvider tree, navigate to the InfoCube whose queries you want to optimize.
2. In the context menu of the InfoCube, choose *Maintain Aggregates*. The *Maintain Aggregates* screen appears. If an aggregate has already been created for the selected InfoCube, you can also get to the maintenance screen by double-clicking on .

Access from Transaction RSDDV

1. On the *Aggregate/BI Accelerator Index: Select InfoCube* screen (transaction RSDDV), select the required InfoCube.
2. Choose *Aggregates.* The *Maintain Aggregates* screen appears.

If you are creating the first aggregate for an InfoCube, the *Proposals for Aggregates* dialog box appears first. You can choose whether the system proposes aggregates or whether you want to create them manually.

For more information, see Creating the First Aggregate for an InfoCube [Page 671].

3. The left side of the screen shows the dimensions, characteristics and navigation attributes of the selected InfoCube in a tree structure as *Selection Options for Aggregates*.

   Select one or more objects to be copied to the aggregate.

   Define the **granularity** you require for the data in the aggregate. Add all the characteristics derived from these characteristics.

   For example, if you define an aggregate for the month, you should also include the *quarter* and *year* in the aggregate.

   This enhancement does not enlarge the dataset, but allows:

   - A year aggregate to be built from this aggregate
   - Those who need the annual values to use the queries for this aggregate.

   You can only include a characteristic and one of its attributes in an aggregate in expert mode (*Extras* → *Switch Expert Mode On/Off*). An aggregate of this type has the same granularity and size as an aggregate that has only been built using the characteristic, but is affected by the hierarchy/attribute change run. Compared with the aggregate for the characteristic in which the attribute information is defined by a join with the master data table, the aggregate for the characteristic *and* the attribute only saves the database join.

   We therefore recommend that you **either** build an aggregate using the characteristic **or** you build a much smaller aggregate using the attribute.

4. You have different options for creating an aggregate:

   - Transfer the selected object(s) to the *Aggregates* column on the right side of the screen using drag and drop.
   - Select *Create New Aggregate*.

   The *Enter Description for Aggregate* dialog box appears.

5. Enter:

   Short description

   Long description

   To change the text at a later time, select *Change Description Text* in the context menu of the aggregate.

6. Choose *Continue*. The *Maintain Aggregates* screen appears.

   The system displays the aggregate in the top-right area of the screen. The log is displayed in the lower part of the screen.

   For more information, see Design and Components of Aggregates [Page 676].

7. If an aggregate contains a time-dependent component, you must assign a key date to the aggregate.

When you fill the aggregate, the key date behaves like the key date of a query: The time-dependent attributes and hierarchies are evaluated on this key date. For this reason, aggregates with a time-dependent component can only be used in a query if the key date of the query is the same as the key date of the aggregate.

In the *Select Variable or Fixed Date* dialog box, select the following as the key date:

A variable that is also used in queries for the key date and can be automatically calculated in the *SAP Exit* or *Customer Exit* processing types (see Variables [External]),

0CWD current workday

0DAT current calendar day

0P_KEYDT key date of due date

0P_KEYD2 key date of posting (from key date of due date)

0P_KEYD3 key date of clearing (from key date of due date)

0P_KEYD4 key date of posting (posting date)

0P_KEYD5 key date of clearing (from key date of posting)

Aggregates with a variable key date must be updated regularly. You have to include this process in a process chain [Page 375] (*Further BI Processes -> Adjust Time-Dependent Aggregates*).

- A particular calendar day

To enter a calendar day, select object CALENDAR (<Calendar>) in the *Select Variable or Fixed Date* dialog box. Choose *Transfer Selection.* The *Calendar* dialog box appears. You can copy this date to the aggregate definition by double-clicking on it.

In the aggregate tree, under *Properties* for node *Variables for Key Date*, the system displays the technical name of the variable you chose or the fixed calendar date.

Once the aggregate has been activated and filled, the system copies the key date computed from the variable for when the aggregate should be filled, into line *Key Date*.

8. By default, the system partitions the aggregate fact tables when the related InfoCube is partitioned and the partitioning characteristic is available in the aggregate. Choose *Properties* → *Change Partitioning* to prevent partitioning for individual aggregates.

If aggregates do not contain much data, very small partitions can result. This affects read performance. Aggregates with very little data should **not** be partitioned. Note that if you change this property to *Not Partitioned* for an existing aggregate, you have to activate and fill the aggregate again.

9. You can change the structure of the aggregate by adding additional components or deleting existing ones. You can also change the key date.

   Inserting components into the aggregate

   i. Select one or more objects in *Selection Options for Aggregates*.

   ii. Use drag and drop to transfer them to the aggregate that you want to change on the right-hand of the screen.

   Where necessary, change the selection type (BW-WHM) [External] by choosing the appropriate entry in the context menu:

   All characteristic values [Page 667]

   Hierarchy level [Page 669]

   Fixed value [Page 669]

   Aggregates containing fewer than 14 components are stored on the database in optimized form (see Loading Data into Aggregates Efficiently [Page 691]).

   Note that the characteristics that are defined in the InfoCube are also included in the aggregate and thus increase the number of components, even though they are not visible on the interface.

   Deleting components from the aggregate

   i. In the aggregate tree, navigate to the characteristic(s) or navigation attributes that you want to delete.

   ii. Select these components.

   iii. To delete the components from the aggregate, you can:

   – Choose *Remove Components* in the context menu.

   – Use drag and drop to copy the components to the left part of the screen.

   To delete a dimension from an aggregate, you have to delete all the characteristics and navigation attributes of this dimension.

   Changing the key date

   i. In the aggregate tree under *Properties*, select the node *Variable for Key Date* and choose *Change* in the context menu. The *Select Variable or Fixed Date* dialog box appears.

   ii. Select and transfer the required variable or calendar day.

   The key date computed from a changed variable is only copied to line *Key Date* if the *Adjust Time-Dependent Aggregates* process has been executed.

10. To check the aggregate definition for inconsistencies, choose *Check Definition*.

11. Save the new or changed aggregate.

Result

You can activate the new or changed aggregate and fill it with data. It is available for reporting.

4.4.5.2.3 Design and Components of Aggregates

Definition

On the *Maintaining Aggregates* screen in the *Display of Aggregates and Their Components* screen area, the aggregates and their components are displayed in a logical tree structure.

Use

This screen area is used to:

- Define aggregates
- Obtain information about the status of individual aggregates

You can define several aggregates for an InfoCube. However, make sure that this is useful:

- Advantage: Aggregates improve the performance of queries
- Disadvantage: Aggregates increase load time
  - when uploading data packages
  - during the hierarchy/attribute change run after loading master data
  - when modifying time-dependent aggregates.

To optimize an InfoCube, you should repeatedly check:

- Whether aggregates are missing: Create new aggregates.
- Whether existing aggregates are still being used: Delete unnecessary aggregates.

The aggregates display in the *Maintenance for Aggregate* screen helps you to evaluate aggregates.

Structure

Column	Information
Aggregates	Aggregates with all transferred components in a tree display
Technical Name	Technical name of the aggregate.
Save	Marks new or changed aggregates
Proposed Action	The system proposes actions, as required. Once the system has proposed a new aggregate, it recommends that you activate this aggregate. This is marked by in this column.
Status	Created Changed (the modified aggregate definition is no longer the same as the active aggregate definition)

	Saved and active
Filled/Switched Off	Not filled with data Filled with data
Individual components in the aggregate (see Selection Type (BW-WHM) [External]):	By default, the system aggregates according to the values of the selected objects: • * All characteristic values • H Hierarchy level • F Fixed value In the context menu of the selection type, you can change this setting.
• *Hierarchies*	Name of the hierarchy you have selected, where applicable.
• *Hierarchy Level*	Level of the hierarchy you have selected, where applicable.
• *Fixed Value*	Fixed value you have selected, where applicable.
Valuation	Representation in a bar chart: • The greater the number of minus signs, the worse the evaluation of the aggregate: "-----" means: Aggregate can potentially be deleted. • The larger the number of plus signs, the better the evaluation of the aggregate: "+++++" means: Aggregate potentially very useful. The evaluation is based on a number of criteria. The following criteria are currently used in the evaluation: • How does the compression of the data compare with the InfoCube: How much smaller is the aggregate compared to the InfoCube? • When was the aggregate last used? (See under *Last Used*).
Records	Number of records in the filled aggregate. This provides information about the size of the aggregate.
Records Summarized (Mean Value)	Number of records read on average from the source to create a record in the aggregate. This example is based on an aggregate with three records. If 10 records are read for the first record in the aggregate, 15 records are

	read for the second record and 20 records are read for the third record, the aggregate has a mean of **15** "compressed records". This provides information about the quality of the aggregate. • The greater the value, the greater the compression and better the quality of the aggregate. Since an aggregate should be 10 times smaller than its source, the number should be larger than 10. • If the value is one, the aggregate is a copy of the InfoCube. In this case, consider deleting the aggregate.
Use	Number of uses (in queries). How often is the aggregate used for reporting?
Last Used	Date When was the aggregate last used in reporting? If an aggregate has not been used for a long time, deactivate or delete it. Note that certain aggregates cannot be used at certain times (for example during vacation). Do not delete basic aggregates that you created to speed up the change run.
Last Roll Up	Date When was data last entered for the aggregate?
Last Roll Up By	User name of the person who scheduled roll up.
Last Changed On	Date When was the aggregate definition last changed?
Last Changed By	User name of the person who last changed the aggregate.

Integration

Log display

When you have defined the aggregate and filled it with data, the log display is shown in the lower part of the screen.

Status overview of aggregates

There is a status display for all the aggregates in the BI system in the *Administration* functional area of the Data Warehousing Workbench:

In the navigation pane, under *Monitors*, choose *Aggregates.* The *Status of the Aggregates* screen area lists all the InfoCubes and all the existing aggregates under each InfoCube.

The structure of this screen area corresponds to the *Aggregate* screen area of the *Maintenance for Aggregate* screen. Only the status of the aggregates is displayed, not their components.

By double-clicking on for a particular aggregate, the *Aggregate Display* screen appears. The aggregate is displayed here together with its components.

4.4.5.2.4 Further Editing Functions for Aggregates

Use

In order to work efficiently with aggregates, you must check the structure and use of the aggregates. You can save time during uploading by either deactivating or deleting the aggregates that you no longer need for reporting.

Functions

Function	Information
Aggregate Tree	The *Aggregate Tree* dialog box appears. The system shows how the aggregates of an InfoCube lie in relationship to one another. In other words, which aggregate can be built from which other aggregate. With the help of the aggregate tree, you can identify similar aggregates and manually optimize the specific aggregates on this basis. Choose automatic optimization in the *Maintenance for Aggregate* screen from *Propose→ Optimize*. For further information about automatic optimization, look under Automatically Selecting and Optimizing Aggregates [Page 680]).
Switch On/Off	You can temporarily switch off an aggregate to check if you need to use it. An aggregate that is switched off is **not** used when a query is executed. To do this select the relevant aggregate and choose *Switch On/Off*. An aggregate that is switched off is marked in column *Filled/Switched off* with . Since aggregates that are switched off must also be consistent, you do not have to activate the aggregate again or to fill it when you switch it back on. Execute a query or trace that would use the aggregate that was switched off. Compare the

	time that the database needs with the time that the query needs when using the aggregate. If the query is not significantly slower without the aggregate, you can deactivate or delete the aggregate. You can find more information about evaluating and controlling the use of aggregates in Displaying Aggregates and their Components [Page 676].
Deactivate	The system deletes all the data and database tables of an aggregate. The definition of the aggregate is not deleted. Select the required aggregate and choose *Deactivate.* The status display in the columns *Status* and *Filled/switched off* change back to . If you want to, you can activate and fill the aggregate again later.
Delete	The system deactivates the aggregate and deletes the definition of the aggregate. To do this, select the aggregate to be deleted and select the delete function either with *Delete* or from the context menu of the aggregate.

4.4.5.2.5 Automatic Selection and Optimization of Aggregates

Use

Data for the BI statistics InfoCube and the created queries can be used as part of the automatic selection and optimizing of aggregates.

In this, the system makes an assessment based on the definition of particularly suitable aggregates. Aggregates that appear to be suitable are proposed automatically. You can optimize the proposed aggregates and use them for further working with the InfoCube.

Prerequisites

An active version of the InfoCube is available.

Before you can use the query statistics, queries have to exist for the InfoCube.

These must be collected before BI statistical data can be analyzed. In order to collect statistical data, the corresponding function has to be activated for the InfoCube (*DW Workbench → Tools → BI Statistics for InfoCubes*) and queries must have already been executed.

Features

You can choose *Proposals* in the menu if you want the system to propose aggregates. You have the following options:

- Proposals from queries: The system considers the queries that are created for an InfoCube.
- Proposals from the previous navigation: The system evaluates the last navigational step that you carried out with a query.
- Proposals from BI Statistics (tables): The system considers BI statistical data (database tables).
- Proposals from BI Statistics (InfoCube): The system considers the data that is contained in the BI Statistics InfoCube.

  You can also postprocess the aggregates and add or delete characteristics.

See also:

Proposals from Queries [Page 681]

Proposals from BI Statistics [Page 682]

4.4.5.2.5.1 Proposals from Queries

Use

By choosing *Proposals* → *Proposals from Queries*, you can select existing BI queries for the selected InfoCubes and view proposals for a minimum and maximum aggregate for each InfoCube. The name of the proposed aggregate is derived from ‘MIN’ or ‘MAX’ and a sequential number.

We recommend that you use this function the **first time** you optimize the InfoCube. If you have already executed queries, use the other options for optimizing, because the number of times a query has been executed and the individual navigational steps are also taken into account.

The **minimal aggregate ‘MIN’** corresponds to the smallest aggregate possible. This only contains the data that is needed for the initial drilldown on a query.

The query has to be set to hierarchical reading or reading on demand.

Activating a minimal aggregate accelerates the drilldown defined in the BEx Analyzer.

The **maximum aggregate ‘MAX’** represents an aggregate that supports every navigational step of a query. This is based on the assumption that you want to perform drilldown for the query using all characteristics (all free characteristics too). Generally this is rarely the case. More frequently, the aim is to be able to use an aggregate of this type in all possible navigation steps of the query. In any case, check whether it makes sense to use the maximum aggregate. This is not the case if the volume of data for an aggregate of this type corresponds to the size of the InfoCube.

If the components (the characteristics and attributes of a newly defined aggregate that are not summarized) are identical to an aggregate that has already been proposed, they will not be added to the list. Instead, the number of calls is increased by one.

Generally the number of calls an aggregate has, the more useful it is: The higher the number of calls for an aggregate, the higher the number of queries in which it can be used.

You can modify or delete the proposed aggregates.

See also:

Proposals from BI Statistics [Page 682]

Optimizing Proposed Aggregates [Page 683]

4.4.5.2.5.2 Proposals from BI Statistics

Use

If you have activated the collection option for query runtime statistical data, every navigational step that requests data from the database is saved. This includes the characteristics, navigation attributes and hierarchies that are involved. First the system saves this statistical data in database tables. You can then load this data into BI statistics for evaluation purposes.

You can also use this statistical data for optimizing aggregates. The advantage of having proposals from BI statistics as opposed to proposals from BI queries is that the actual user capacity is taken into account here.

We recommend that you use this function if representative statistical data already exists. You run this function at regular intervals to modify the aggregates in accordance with changes to user actions.

You can evaluate the data saved in BI Statistics (database) or BI Statistics (InfoCube) by selecting the menu path *Proposals* → *Proposals from BI Statistics (Tables)* or *Proposals from BI Statistics (InfoCube)*. You can restrict the analysis to a subset of the data by specifying an interval for the start time or runtime of the query.

After the data has been read from BI Statistics, the optimal aggregate is determined for every navigation step, and a list of the different aggregates is created.

For aggregates in a component that vary only in terms of their selection type, all aggregates with the selection type hierarchy (‘H’) or fixed value (‘F’) are replaced with an aggregate that is grouped by characteristic values (‘*’). This is possible as long as the aggregate has already been proposed.

The list of proposed aggregates has the same structure as the proposals from BI queries. However, the list is generally longer.

You can modify or delete the proposed aggregates.

See also:

Proposals from Queries [Page 681]

Optimizing Proposed Aggregates [Page 683]

4.4.5.2.5.3 Optimizing Proposed Aggregates

Use

If the analysis of BI queries or BI statistics has proposed several aggregates, it is not advisable to activate all of them. Using aggregates reduces the runtimes of queries and the roll up of data is optimized in such a way that allows aggregates that are already rolled up and available to be used. But, as a rule, the total memory space required for all aggregates is too great and it takes too long to fill the aggregates.

In addition to the runtime for the queries, a complete optimization must also take into account the dependencies of the aggregates, their memory requirements, the time taken to roll up new data, and other factors.

Activities

You can run a simplified optimization by choosing *Proposal → Optimize.*

For this optimization it is usually assumed (heuristics), that the number of aggregates should be reduced first. In addition, those aggregates are selected that have been called up least often and together account for 20% of all calls. These aggregates are checked, one after the other, to see if there are any aggregates with exactly one extra component.

If the system finds more than one aggregate with exactly one additional component, it chooses the aggregate that has been called the most number of times. The calls for the aggregates that have been checked are added to this number. The checked aggregate (from the 20% quantity) is then deleted from the list of proposed aggregates.

However, this only happens if the number of calls for the checked aggregate is not more than double the calls for the aggregate with the extra components. This prevents aggregates from being replaced by others that are used relatively rarely.

You can continue to optimize until the point where the aggregate is small enough, or until no more aggregates can be compressed.

Since the optimizer has no information about the data structure, you should check the proposals again before filling aggregates with data. For example, a proposed aggregate may contain a characteristic that would make the aggregate almost the same size as the InfoCube. This would mean that when the aggregate is filled, the system virtually creates a copy of the InfoCube. This is not generally the objective when using aggregates.

See also:

(See System Response Upon Changes to Master Data and Hierarchies [Page 689])

4.4.5.3 Activation and Provision of Data

Use

The following sections explain how you activate aggregates and fill them with data.

- If you have created or changed an aggregate, you have to activate it and fill it with data before you can use it when you execute a query. See Activating and Filling Aggregates [Page 684].
- If you have loaded new data packages (requests) to the InfoCube, you have to roll up these data packages into aggregates before the current data is available for reporting on the aggregate. See Rolling Up Data in Aggregates [Page 687].
- If the hierarchies and attributes of characteristics that belong to an InfoCube have been changed, you have to make structural changes to the aggregates to modify the data accordingly. See System Response Upon Changes to Master Data and Hierarchies [Page 689].
- You can make the data load more efficient both when initially filling the aggregate and during roll up. See Loading Data into Aggregates Efficiently [Page 691].

4.4.5.3.1 Activating and Filling Aggregates

Use

To use an aggregate for an InfoCube when executing a query, you must first activate it and then fill [External] it with data.

Prerequisites

You have created or changed an aggregate.

Procedure

1. Select the aggregate that you want to activate and fill.
2. Choose *Activate and Fill*. The system creates an active version of the aggregate.

System Activity	Result
The system creates the tables required by the aggregate definition in the database. Aggregates are created according to the same schema as InfoCubes. • An aggregate contains two fact tables and a number of dimension tables. • The table names are derived from the technical names of the aggregates.	If the aggregate was activated successfully, the status display in the *Status* column changes from for a newly create aggregate or for a changed aggregate to .

3. Once the aggregate was activated, you must trigger the action to fill the aggregate with data. The *Subsequently Aggregate the Aggregates of an InfoCube* dialog box appears.
4. An active aggregate is selected and marked in the *Active/Inactive* column with an .

   If you do **not** want to fill an aggregate, you can deselect it.
5. Select *Fill aggregate*. The *Execution Time of Aggregation* dialog box appears.

   Since it can take a long time to build an aggregate from an InfoCube, all the aggregates are filled in the background.

   Note that an aggregate can also read data from a larger aggregate that is already filled. You can therefore assign data to very compressed aggregates quickly.
6. Define when you want to start the job to fill the aggregate:
   - *now*
   - *later*

     This takes you to the *Time of Subsequent Aggregation* dialog box. Enter the date and time for the background processing.
7. In the *Subsequently Aggregate the Aggregates of an InfoCube* dialog box, choose *Refresh.* The system copies the relevant data to the columns
   - User who scheduled the aggregate for building,
   - Date for scheduling the new version of the aggregate
   - Time for scheduling the new version of the aggregate
   - Name of the background job

   In the column *Job Status* you have the following display options:

   shows that the job of filling the aggregate is currently running.

   shows that the job of filling the aggregate is scheduled, but is not running yet.

   shows that the job of filling the aggregate was canceled.

   To find out about your background jobs for filling aggregates, choose *Jobs*. You get to the *Simple Job Selection* screen.

   To view the logs for filling your aggregates, choose *Log*. The *Evaluate Application Log* dialog box appears.

   By using the transaction SLG1, you can directly access the application log even if the job is not canceled. The *Evaluate Application Log* dialog box appears.

   Enter the required data in the following fields:

Field labels	**Entry**
Object	RSSM (Scheduler; Monitor; Tree Callback)
Subobject	MON

	(Monitor)
Ext. Identif. (External Identification)	For displaying all logs: **InfoCube** For filling new aggregates: MON:PROTOCOLL_ACTION-AGGR2-*InfoCube* (For rolling up already existing aggregates: MON:PROTOCOLL_ACTION-AGGR1-*InfoCube*) Instead of entering *InfoCube*, enter the technical name of the desired InfoCube.

8. Choose ✔.

System Activity	Result
The system reports all the executed actions in the lower right part of the log display. If you used a variable for the key date in an aggregate with time-dependent attributes or hierarchies, the system evaluates this variable when it fills it and builds the aggregate on the computed key date.	If the aggregate was filled successfully, the status display in the column *Filled/Switched off,* changes from 🔴 to 🟢.

To activate and fill a number of aggregates at the same time, select them and choose *Activate and fill*.

The system chooses the optimum sequence for filling the aggregates.

i. The large, detailed aggregates are filled first.

ii. The smaller, very compressed aggregates are filled next.

The larger aggregates can therefore be used already when you are still building the smaller ones.

Result

The active aggregate that is filled with data can be used for reporting. If the aggregate contains data that is to be evaluated by a query, the query data is read automatically from the aggregate.

You can find more information about the number of records read and the use of an aggregate in queries in Displaying Aggregates and their Components [Page 676].

You can roll up new data packages (requests) into the aggregate. For more information see Rolling up Data into an Aggregate [Page 687].

4.4.5.3.2 Rolling Up Data in Aggregates

Use

When you load new data packages (requests) into the InfoCube, these are not immediately available in reporting for use in an aggregate. To fill the aggregate with the new data from the InfoCube, you first have to load these into the aggregate tables for a set time frame. This process is known as roll up [External].

Prerequisites

- New data packages (requests) have been loaded into an InfoCube.
- Aggregates for this InfoCube have been activated and filled with data.

Procedure

In InfoCube maintenance, you can specify how the data packages are rolled up in the aggregate. You can do this on an InfoCube-by-InfoCube basis.

You are in the Data Warehousing Workbench in the *Modeling* functional area. In the context menu of the required InfoCube, choose *Manage*.

The system copies the InfoCube data into the table at the top of the screen.

To optimize data load performance, you can specify that you want to automatically delete indexes before the load operation and recreate them when the data load is complete. You do this on the *Performance* tab page. Building indexes in this way accelerates the data load process, although it has a negative impact on system performance when the data is read. Only use this method if no read process takes place during the data load.

If you want to switch on index building during roll up anyway, choose *Create Index (Batch)* and select the required options: *Delete InfoCube Indexes Before Each Data Load and then Refresh* or *Also Delete and then Refresh Indexes with Each Delta Upload*.

Type of Execution	Procedure
Including roll up of data package as a process in a process chain We recommend that you include this in a process chain • with complex schedules • where there are difficulties with automations and event collectors • for all new development	1. Choose the *Roll Up* tab page. The system proposes the greatest possible value for *Roll Up to Request ID*. However, you can overwrite this. 2. Choose *Process Chain Maintenance*. Transaction RSPC takes you to the *Process Chain Selection* dialog box. Here you obtain an overview of the process chains that exist in the BI system. If no suitable process chain is available, you can create a new process chain for the

	roll up. For more information, see section *Creating Process Chains by Using a Maintenance Dialog for a Process* in Creating Process Chains [Page 402].
Starting data package roll up manually Use this procedure if the data from several data packages creates a logical unit, and therefore only needs to be released. Different plants deliver their data at different points in time. The data only needs to be visible in the InfoCube when all plants have loaded their data into the InfoCube.	1. Choose the *Roll Up* tab page. 2. Choose *Selection*. The *Start Time* dialog box appears. 3. Choose the start time. You have the following options: – Immediate start – Date/time – After job – After event – By operation mode 4. Set the appropriate indicator if you want to run the job periodically. 5. Save your entries.

The following procedures are also possible but should not be used for new scenarios.

Type of Execution	Procedure
Subsequent Processing [External]	1. Choose the *Roll Up* tab page. 2. Choose *Postprocessing*. The *Maintain Events to Run After Processing* dialog box appears. For more information, see Process Control [External] and Creating an Event Collector [External].
Roll up each data package in the aggregate automatically The InfoCube has to be technically correct and the quality has to be assured. Only use automatic roll up if you are loading requests into the InfoCube and there is no overlap between the processes for loading, roll up, and the other automatisms for the InfoCube. For more information, see Automatic Subsequent Processing [External].	1. On the *Manage Data Targets* screen choose *Environment -> Automatic Request Processing*. The *Automatism Maintenance* dialog box appears. 2. Under the group header *Automatic Processing*, set the *Roll Up Data into the Aggregate* indicator. 3. Save your entries.
Program RSDDK_AGGREGATES_ROLLUP	You can also run roll up using program RSDDK_AGGREGATES_ROLLUP (see Direct

	Execution - Reports [External]). You can schedule this program as a regular background job or use it in an Event Collector [External].

Result

The new data is available in reporting for queries that are started after the roll up.

See also:

Managing InfoCubes [External]

4.4.5.3.3 System Response Upon Changes to Master Data and Hierarchies

Use

If the hierarchies and attributes of characteristics that belong to an InfoCube have been changed, you have to make structural changes to the aggregates to modify the data accordingly.

Note that with a structural change, **all** aggregates of **all** InfoCubes are modified if they are affected by the changes to the hierarchies and InfoObjects. This may take some time.

You can still report on the ‘old’ hierarchies and attributes during the change run.

Integration

If the changes affect an amount of data that exceeds a reasonable limit, modifying the aggregate is more time consuming than reconstructing it. You can change this threshold value. In the Implementation Guide (IMG), choose *SAP NetWeaver* → *Business Intelligence* → *Performance Settings* → *Parameters for Aggregates* in the section **Percentage Change in the Delta Process**. In the *Limit with Delta* field, enter the required percentage (a number between 0 and 99). 0 means that the aggregate is always reconstructed. Change these parameters as many times as necessary until the system response is as quick as possible.

Features

If an aggregate is affected by changes to the data, it is either modified (in a delta process) or reconstructed. When you modify an aggregate, the obsolete data records are read negatively whereas the new data records are read positively.

You can modify aggregates manually or automatically using a program.

You can start multiple change runs simultaneously. The prerequisite for this is that the lists of master data and hierarchies to be activated are different and that the changes affect different InfoCubes. If a change run terminates, you have to start the same change run again. You do

this by starting the change run again with the same parameters (same list of characteristics and hierarchies).

Activities

Manual Modifications

1. In the Data Warehousing Workbench menu, select *Tools* → *Hierarchy/Attribute Changes*. (Alternatively, in the Data Warehousing Workbench, in the *Administration* functional area, choose *Monitors* → *Change Run*). The **Execute Hierarchy/Attribute Changes for Reporting** screen appears. On this screen, all the executed change runs are listed with detailed information. All the InfoObjects and hierarchies that are scheduled for the structural change are selected by default.
2. If you only want to carry out the structural changes for individual InfoObjects and hierarchies, select the *InfoObject List* or *Hierarchy List* pushbutton. Remove from the list the InfoObjects and hierarchies that you do **not** want to change.
3. Schedule a new structural change by choosing **Selection** and determine the start date. The status of the structural change is displayed in the upper section of the screen.

Automatic Modifications

To schedule the same function when executing a report directly [External], enter the program name **RSDDS_AGGREGATES_MAINTAIN**.

You can also include the program in a process chain and schedule it regularly in background processing.

The attributes for characteristic 0MATERIAL are uploaded weekly. You can schedule the program so that it starts after the upload. You can also use InfoObject List 0MATERIAL as a variant so that only changes that are made to the material attributes are taken into account. Other changes, which may only be needed later, are ignored.

Choose *Log* to display the messages for the hierarchies and attributes that have been changed manually or automatically.

Status Overview of Change Run

In the *Administration* functional area of the Data Warehousing Workbench, you find a status display for all executed change runs in the BI system. In the navigation pane, choose *Monitors* → *Change Run.* The *Monitor for the Change Run* screen appears.

Choose *Change Run*. A dialog box for selecting a run ID appears.

Choose *Logs*. A dialog box for selecting a process type appears. Choose *Change Run.* The screen for the display of the relevant log in the application log appears.

4.4.5.3.4 Loading Data into Aggregates Efficiently

Setting Automatic Compression

For each InfoCube, you can set whether the aggregates of the InfoCube are compressed automatically when it is filled with data or after the roll up of data packages (requests).

1. You are in the *Data Warehousing Workbench* in the *Modeling* functional area. In the InfoProvider tree, navigate to the required InfoCube.
2. In the context menu of the InfoCube, choose *Manage*.
3. In the lower part of the screen, select the *Roll Up* tab page.
4. Under the *Aggregates* group header, set the corresponding indicator in the *Compress After Roll Up* field.

   Alternatively you can set automatic compression after roll up. This is described below:

   You are in the Data Warehousing Workbench in the *Modeling* area. In the context menu of the required InfoCube, choose *Display* or *Change*. Choose *Environment* → *InfoProvider Properties* → *Display* or *Change*. On the *Roll Up* tab page, choose option *Compress After Roll Up*.

Indicator	Information
Set (default): Automatic compression switched on	The aggregates of an InfoCube are compressed automatically when the InfoCube is filled with data or after the roll up of data packages (requests). If you want to delete a data package (request) from the InfoCube and the InfoCube has already been rolled up to the aggregate, you have to deactivate the aggregate and build it again.
Not set: Automatic compression switched off	The aggregates are only compressed with the InfoCube. Use this setting if you have to frequently delete requests from the InfoCube. A specific request can be deleted from the aggregates when it has been deleted from the InfoCube. Note the possible affects on performance; aggregates can become quite large if they are not compressed automatically.

Reading the Data in Blocks

If the amount of data is very large when you fill the InfoCube, the system reads the data in blocks and not all at one time. This avoids problems with temporary table space on the database which may occur if you have very large sources (InfoCubes or aggregates).

For more information about the block size settings, see Customizing under *SAP Customizing Implementation Guide* → *SAP NetWeaver* → *Business Intelligence* → *Performance Settings* → *Parameters for Aggregates*.

Optimizing Performance of Aggregates with Fewer Characteristics

Aggregates with fewer than 14 characteristics are created for all databases in such a way that each characteristic is in a separate dimension (artificial) and these dimensions are created as line item dimensions. Aggregates that only consist of line item dimensions are filled purely from the database. This improves the performance when filling and rolling up.

The logical tree display in the right part of the *Maintenance for Aggregate* screen is copied from the left part of the *Selection Options for Aggregates* screen but does not mirror this special form of storage on the database.

Optimizing Data Load Performance

To optimize data load performance, you can specify that you want to automatically delete indexes before the load operation and recreate them when the data load is complete. Building indexes in this way accelerates the data load process, although it has a negative impact on system performance when the data is read. Only use this method if no read process takes place during the data load.

If you want to switch on index building during roll up anyway, you have the following options:

- You are in the Data Warehousing Workbench in the *Modeling* area. In the context menu of the required InfoCube, choose *Display* or *Change*. Choose *Environment* → *InfoProvider Properties* → *Display* or *Change.* On the *Database Performance* tab page, chose options *Delete index before each data load and then recreate* or *Delete index before each data load and then recreate*.
- You are in the Data Warehousing Workbench in the *Modeling* area. In the context menu of the required InfoCube, choose *Manage*. On the *Performance* tab page, choose the option *Create Index (Batch)* and select the required options: *Delete InfoCube Indexes Before Each Data Load and then Refresh* or *Also Delete and then Refresh Indexes with Each Delta Upload*.

Parallel Execution of Processes for More Than One Aggregate

In *BI Background Management* (transaction RSBATCH), you can specify that the following processes are executed in parallel. These processes all serve to process aggregates. Parallel processing is applied to the aggregates in any number of InfoCubes.

Process types for parallelization

Process Type	Description
AGGRFILL	Initial filling of aggregates
ATTRIBCHAN	Attribute change run
CHECKAGGR	Check aggregates during roll up
CONDAGGR	Compress aggregates
ROLLUP	Roll up

For **roll up** you can also make these settings in the InfoCube:

You are in the Data Warehousing Workbench in the *Modeling* area. In the context menu of the required InfoCube, choose *Manage*. On the *Roll Up* tab page, choose *Parallel Processing*. A dialog box appears in which you can define settings for parallel processing.

For the **change run**, you can also make the settings in the *Administration* functional area of the Data Warehousing Workbench. Go to *Change Run*: Under the group header *Executing Change Runs*, choose *Parallel Processing*. A dialog box appears in which you can define settings for parallel processing.

By default, the system executes a maximum of three parallel processes. You can change this setting (*Number of Processes*) for each individual process type. In process chains, the affected setting can be overridden for each of the processes listed above.

Note that fill, roll up and change run each consist of several subprocesses, all of which are processed in parallel.

For example, roll up consists of the following subprocesses:

- Roll up data into an aggregate
- Compress, as required
- Check, as required

The parallel processing settings for the subjobs correspond to the parallel processing settings for the main job. For example, if you decide that you want to perform roll up in five parallel processes and compression in two, the system executes the compress subprocess of the roll up in five parallel processes.

If you do **not** want the system to respond in this way, you can set parameters for the InfoCube so that the system does **not** automatically compress the aggregate (see section *Setting Automatic Compression* above). In addition, you can add the *Compress Aggregate* process as a subsequent process to the *Roll Up* process in a process chain. In this case, the system applies the compression settings that you set in the *BI Background Management* transaction (transaction RSBATCH). In the example above, the system executes roll up in five parallel processes and compression in two.

The parallel processes are executed in the background, even if the main process is executed in the dialog. This can considerably decrease execution time for these processes. You can determine the degree of parallelization and specify the server on which the processes are to run and with which priority (job category). Job category ’A’ has the highest priority, followed by category ’B’ and finally ’C’.

Note that if you choose more than two parallel processes (*Number of Processes*), one process monitors the other processes and divides the work packages. You always have one process less in actual usage than the number of processes selected in the settings.

4.4.5.4 Performing Checks for Aggregates

Use

You perform aggregate checks to check the correctness of the data records in an aggregate. You can define as many aggregate checks as you require for each InfoCube. You can check the required aggregates in different check modes at any time.

You can use check aggregates or strict characteristic restrictions to run quick aggregate checks on a daily basis after roll up and, in addition, schedule a complete check of critical aggregates at the weekend. Or you can run checks on a weekly basis after weekly change runs.

You can view the results of the check in the logs in the application log [External]. If the system finds incorrect records, it saves these incorrect records in a new database table (/BI0/01xxxxxx). You can see the name and size of the table in the log messages.

You **cannot** transport aggregate checks because the number, type and size of the aggregates in the test and productive systems are normally different. For this reason, you need to create individual aggregate checks in each relevant system. You can create analogous checks with identical check IDs in the different systems.

Procedure

You access aggregate check maintenance either from aggregate maintenance (transaction TSDDV) in the *Maintaining Aggregates* screen (choose *Extras* → *Automatic Check (On/Off/Change)*), or from transaction RSDDAGGRCHECK. The *Maintain Aggregate Check: Select InfoCube* screen appears.

Selecting Checks

Each check is specified by the InfoCube name and an ID.

1. Enter the name of the InfoCube.
2. Enter a valid check ID:
   - If you want to display, exit, execute or delete a check that already exists, enter the corresponding check ID. Input help is available for both the InfoCube name and the check ID.
   - If you want to create a new check, provide an (unused) check ID. If you leave the field empty, the system automatically generates a new check ID.

Editing Checks

You can create, change, execute, execute ad hoc or delete aggregate checks for an InfoCube.

The following functions are available:

Editing Function	Description

Display	The *Display of Check Time for the Aggregate* screen appears. The system displays the aggregate tree of selected aggregates with their check modes and check times. If characteristic restrictions have been defined for the check for one or more aggregates, these are displayed subsequently in a dialog box. Choose confirm to close the display.
Edit	The *Check-Time Selection for Individual Aggregate* screen appears. The system displays the description of the check with the check mode settings, check time, and characteristic relationships. You can change these properties here. If you save the check, the previous settings are overwritten.

Create	The *Check-Time Selection for Individual Aggregate* screen appears. The system displays the aggregate tree with all the aggregates in the InfoCube. 1. Select the aggregates that you want to check. 2. Determine the check time by setting the corresponding indicator. 3. In the context menu of the aggregates to be checked, choose the check mode. The system displays the selected check mode in the corresponding column. If you choose the *Selection Options* check mode, the *Characteristic Restrictions When Checking Aggregates* screen appears. Enter the restrictions and choose *Continue*. 4. Choose *Continue*. The system checks whether new aggregates have to be created or whether check aggregates exist that are no longer needed because the aggregates have been deleted. The *Confirmation of Aggregate Check* screen appears. The corresponding information is displayed in the *Check Overview* area. 5. Enter the following general check parameters: Description/texts (*short description*, *long description*) for the check. Block size for check: This value is not the block size that is observed when building aggregates but the mechanism behaves in the same way: It ensures that the system can also check large aggregates, without this causing problems regarding temporary table space. 6. Save the check. If check aggregates need to be created for this check, the corresponding aggregates and tables are created and activated first. The dialog box for filling check aggregates appears (see Activating and Filling Aggregates [Page 684]). Schedule a job for filling the aggregate. If you have *Schedule* as the check time in the check aggregate, the dialog box for scheduling in background processing appears. Here you can enter the time(s) or choose *Cancel* if you have only created the check for ad hoc purposes. The system does **not** check an aggregate unless all the check aggregates that this check requires are filled when the check starts. If you have selected aggregates with check time *Now*, the system executes this part of the check in dialog mode and displays the results afterwards in the application log. The settings for aggregates with check time *Now* are **not** transferred to the definition of the aggregate check.

Delete	If you confirm the confirmation prompt, the check is deleted.
Execute	The check you have chosen is executed in dialog mode. The system displays the results in the application log.
Ad hoc	The *Check-Time Selection for Individual Aggregate* screen appears. 1. Select the aggregates that you want to check. 2. Choose *Now* as the check time. 3. In the context menu of the aggregates to be checked, choose the check mode. You can only select the *Check Aggregate* check mode if a corresponding check aggregate already exists. Choose confirm again and the system executes the check you have selected in the dialog.

Evaluating Logs

If the check is executed *After Change Run*, *After Roll Up* or *After Deletion*, you can find the logs for the aggregate check with the logs for the main process.

If you execute an ad hoc check or execute a check using *Now*, the system displays the application log automatically at the end of the check.

If the check is scheduled for background processing and is executed in the background, the logs are available in the application log under object RSRV, subobject AGGRCHECK. The InfoCube name and check ID are recorded in the identifier.

You can also display these logs by choosing *Logs* on the *Maintain Aggregate Check: Select InfoCube* screen.

For more information, see:

Check Time [Page 697]

Check Mode [Page 698]

Technical Information About Aggregate Check [Page 699]

4.4.5.4.1 Check Time

Definition

The check time is the time at which an aggregate is checked. Users determine check times.

Use

You can determine the check time in the following ways:

Check Time	Description

After Change Run	The aggregates are checked immediately after the change run. The system only checks aggregates if they have been modified in previous change runs and switches on the check for these aggregates.
After Roll Up	The aggregates are checked immediately after the data from the InfoCube has been rolled up. The system only checks those aggregates for which the check is switched on.
After (Request) Deletion	See *After Roll Up*.
Schedule	You specify that you want to execute the check at a particular time or periodically in background processing. If you want to execute a particular check frequently but not on a regular basis, choose the *Schedule* check time when you create the check. Save the check. If you cancel the scheduling you can execute the check at any time using program RSDDK_CHECK_AGGREGATE_CHECKID in the dialog or in background processing (see section *As Part of a Process Chain* below).
Now	The check is started immediately as a dialog process. Aggregates that are checked with check time *Now* are **not** included in the definition of an aggregate check and are not saved.

As Part of a Process Chain

You can also define a check in transaction RSDDAGGRCHECK and integrate this check into a process chain. To do this, you have to include program RSDDK_CHECK_AGGREGATE_CHECKID in the process chain with the corresponding check ID and the name of the InfoCube. The system checks all aggregates that have been selected with check time *Schedule*.

Aggregate checks **cannot** be transported. If you have included a check in a process chain and you transport this process chain, there is the possibility that an incorrect check may be executed. In different systems, create checks that correspond to one another with identical check IDs.

4.4.5.4.2 Check Mode

Definition

The check mode determines how an aggregate is checked.

Use

You have the following options for checking the aggregates of an InfoCube. The following table provides an overview.

Check Mode	Description

Full	For the full check, the system builds the aggregate again from the InfoCube as an internal table and compares it with the data of the aggregate in the database, record by record. This check can take a lot of time, but it offers the highest level of security.
Selection Options	For the restricted check, you can set restrictions for characteristics on the *Characteristic Restrictions When Checking Aggregates* screen. You can only select those characteristics that exist in the aggregate and do not have hierarchies or fixed value restrictions defined for them. The system runs the check in the same way as it runs the full check, but only for InfoCube or aggregate data that meets the defined restrictions. Depending on how strict the restrictions are, this check can be considerably quicker than the full check. This type of check is particularly useful in the following case: If the data in the aggregate only changes in a particular time frame, you can restrict the dataset considerably by restricting the check to this time frame.
Aggregated	For the aggregated check, the system aggregates all the characteristics in the InfoCube and aggregate and compares the result of each key figure. This check is 3-4 times quicker than the full check but does not provide the same level of security. You **cannot** perform this check for the following aggregates: • Aggregates of non-cumulative InfoCubes • Aggregates with fixed values • Aggregates with a hierarchy that is not unique or a hierarchy that does not have non assigned nodes.
Check Aggregate	For the check with check aggregates, the system creates a check aggregate that is aggregated using all characteristics. A check aggregate of this type is created for each fixed value combination that occurs in the aggregates you have selected. The check aggregates are filled from the InfoCube and are always modified during roll up or deletion. The check checks that the key figure totals agree. This check is very quick, but it cannot find every potential inconsistency in the aggregates. You **cannot** create check aggregates for the following aggregates: • Aggregates of non-cumulative InfoCubes • Aggregates with fixed values for a navigation attribute • Aggregates with a hierarchy that is not unique or a hierarchy that does not have non assigned nodes.

4.4.5.4.3 Technical Information

The technical information for the aggregate checks is stored in tables RSDDAGGRCHECKDIR, RSDDAGGRCHECKSEL and RSDDAGGRCHECKT.

Class CL_RSDDK_AGGR_AUTOCHECK and function group RSDDK2_CHECK contain the program source code.

The following programs are available for checking aggregates:

- RSDDK_CHECK_AGGREGATE checks as many aggregates as required and also checks aggregates from different InfoCubes in modes A (**All**), Q (**Aggregated**) and C (**Check Aggregate**)
- RSDDK_CHECK_AGGREGATE_SELOPT serves to enter characteristic restrictions and the subsequent aggregate check.
- RSDDK_CHECK_AGGREGATE_CHECKID executes specific checks that have been defined in transaction RSDDAGGCHECK.

4.4.6 Performance Optimization with BI Accelerator

Purpose

BI accelerator (BIA) allows you to improve the performance of BI queries when data is read from an InfoCube. The data of a BI InfoCube is made available as a BI accelerator index in a compressed but not aggregated form.

BI accelerator is particularly useful in cases where relational aggregates (see Performance Optimization with Aggregates [Page 665]) or other BI-specific methods of improving performance (such as database indexes) are not sufficient, are too complex, or have other disadvantages.

For example, if you have to maintain a large number of aggregates for one particular InfoCube, you can use the BI accelerator to avoid this high maintenance effort. Unlike performance optimization with aggregates, there is only **one** BI accelerator index for each InfoCube. Like performance optimization with aggregates, you do not have to make any decisions with regard to modeling for the BI accelerator index.

Implementation Considerations

The BI accelerator is based on TREX technology. To use the BI accelerator, you need an installation based on 64-bit architecture. Hardware partners deliver this variant in preconfigured form as the *BI accelerator box*. Note that you **cannot** use a TREX installation configured for searching in metadata and documents with the BI accelerator since TREX installation is based on a 32-bit architecture. Equally, a BI accelerator box **cannot** be used for searching in metadata and documents. If you want to use the search function as well as the BI accelerator, you require two separate installations.

For more information, see the following SAP Notes:

883726 TREX 7.0: Central Note SAP NetWeaver BI accelerator

902533 TREX 7.0: HowToGuide Connecting/Operating BI Accelerator Box

Integration

Accessing the OLAP Processor

You can use relational aggregates and a BI accelerator index for the same InfoCube. A BI query always tries to use performance-optimized sources by checking the sources from which it can draw the requested data. It checks the sources in the following order:

1. OLAP cache

2. BI Accelerator Index
3. Relational aggregates from the database
4. InfoCubes from the database

If an active BI accelerator index exists, the OLAP processor always accesses this BI accelerator index and not the relational aggregates. Therefore, with regard to modeling, we recommend that you create either relational aggregates or a BI accelerator index for an InfoCube.

Query Execution

When the query is executed, it is clear to the user whether data is being read from an aggregate, a BI accelerator index, or an InfoCube.

In the maintenance transaction, you can deactivate a BI accelerator index on a temporary basis to test it for performance purposes or to analyze data consistency.

You can also execute the relevant query in the **query monitor** (transaction RSRT) using a corresponding debug option: Choose *Execute + Debug*. *In the Debug Options dialog box, choose Do Not Use BI Accelerator Index* to execute the query with aggregates or an InfoCube.

4.4.6.1 BI Accelerator Index

Definition

A BI accelerator index is a redundant data store of a BI InfoCube on the BI accelerator server.

Use

BI accelerator enables quick access to any data in the InfoCube with low administration effort and is especially useful for sophisticated scenarios with unpredictable query types, high volumes of data and a high frequency of queries.

Structure

BI Accelerator Index

A BI accelerator index contains all the data of a BI InfoCube in a compressed but not aggregated form. The BI accelerator index stores the data at the same level of granularity as the InfoCube.

It consists of several, possibly split indexes that correspond to the tables of the enhanced star schema and a “logical” index which, depending on the definition of the star schema, contains the metadata of the BI accelerator index.

BI Accelerator Server

The BI accelerator server is a TREX system as an installation of a BI accelerator engine. The data of the BI InfoCube is kept and processed entirely in the main memory of the BI accelerator server.

The **BI accelerator engine** is the part of the analytics engine that manages the BI accelerator index. This software allows the system to read data from the BI accelerator index, add data to the BI accelerator index, or change data. The **BI accelerator optimizer** is the part of the BI accelerator engine that ensures the best possible read access to a BI accelerator index. For more information, see Technical Information About the BI Accelerator Engine [Page 702].

Integration

Maintenance Processes for BI Accelerator Indexes

With the **BIA index maintenance wizard** you can create, activate, fill and delete BI accelerator indexes.

Like relational aggregates, a BI accelerator index is a redundant downstream data source that is used to improve query performance. For this reason, hierarchy and change run processes and processes for rolling up data are derived from aggregate maintenance. For more information, see Rolling Up Data in BI Accelerator Indexes [Page 710] and System Response Upon Changes to Data: BI Accelerator Index [Page 711].

BI Accelerator Index as InfoProvider for Reporting

At query runtime, analytical engine functions such as aggregation, filtering, selection and some cell-based sorts are performed on the BI accelerator server.

4.4.6.1.1 Technical Information About the BI Accelerator Engine

Processing Data

After creating a BI accelerator index, the data is available on the file server of the BI accelerator server. The data is loaded to the main memory when you execute a query for the first time or start a special load program. The data remains in the main memory until it is replaced or is removed from the main memory when a special delete program is started. It may be necessary to execute a special delete program if, for example, there is not enough memory on the BI accelerator server for all BI accelerator indexes and you need to load data from particular InfoCubes but data from other InfoCubes is not needed (at this time).

Table data is stored in the main memory in columns. Vertically segmenting data tables in this way is more efficient than saving row-based data in conventional relational database systems. In a conventional database, the system has to search all the data in the table if a predefined aggregate is not available for a query. The BI accelerator engine specifically accesses only those data columns that are relevant. It sorts the columns individually and puts the required entry at the beginning. This improves performance considerably because the data flows are smaller. It also significantly reduces the input and output load and the main memory consumption.

Compressing Data

Data is available on the BI accelerator server in a read-optimized format. The BI accelerator engine uses dictionary-based compression. Integers are used to represent text or values in table cells. Using integers allows efficient numeric coding and intelligent caching strategies.

For example, if a column has a thousand rows and some of the cells contain long texts, efficiency is significantly increased by using a ten-bit binary number to identify the texts during processing and a dictionary to call them again afterwards. The datasets that have to be transferred and temporarily stored during the different processing steps are reduced on average by a factor of ten.

This means that you can perform the entire query processing in the main memory and reduce network traffic between separate landscapes.

Divided (Split) Indexes

The BI accelerator engine can process huge datasets, without exceeding the limits of the installed memory architecture. You can split large tables (fact tables and large X and Y tables) horizontally, save them on different servers and process them quickly in parallel. The maximum table size before the system splits the index depends on the existing hardware of the BI accelerator server. Data is distributed to the subindexes in a round-robin procedure. Write, optimize and read accesses are parallelized on the BI accelerator server.

This scalability allows users to make use of sophisticated adaptive computing infrastructures such as blade servers and grid computing.

Index Types

The following index types are available:

- **Normal**: In standard cases, the system creates BI accelerator indexes on the BI accelerator server for all the tables in the InfoCube star schema.
- **Flat**: An exception arises if the InfoCube star schema has been deconstructed because, for example, one (or more) dimension tables have got very large (> 20% of the InfoCube). In this case, the system does not create dimension tables but de-normalizes the appropriate part of the InfoCube star schema (fact and dimension tables).

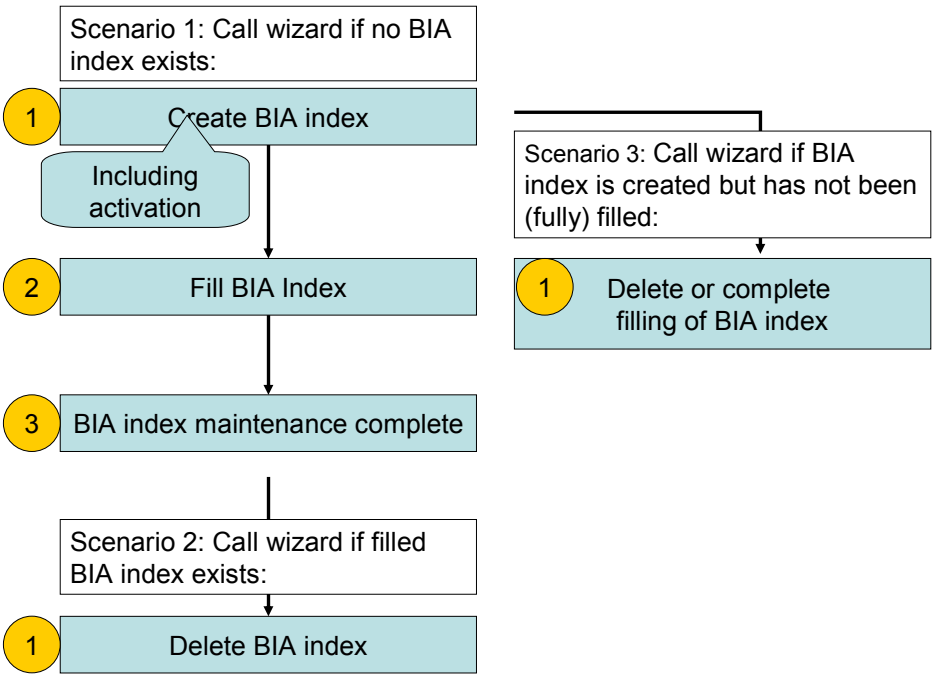
4.4.6.2 Using the BIA Index Maintenance Wizard

Purpose

You can create only one BI accelerator index for each InfoCube. This BI accelerator index contains all the data from the InfoCube. In contrast to the procedure with aggregates, it is not necessary to make specific selections and restrictions for the definition of the BI accelerator index.

Overview of Steps

Using the BIA index maintenance wizard, you can create and activate a BI accelerator index in a step and fill it or delete it as necessary. The following figure outlines the possible steps:



Structure of the BIA Index Maintenance Wizard

In the upper area of the screen, the system displays a text field with information about each step. Corresponding pushbuttons are available:

Function Key	Meaning
Exit Maintenance	You exit the BIA index maintenance wizard.
Continue	You navigate to the next step.

In the center area of the screen, you find the following tab pages:

- On the *Information* tab page, you find additional information about the step.
- On the *Messages* tab page, the system displays information about the current status.
- On the *Index Information* tab page, the system displays the tables or indexes of the BI accelerator index and its properties (see BI Accelerator Index Design [Page 722]).

In this area of the screen, you find the following function keys:

Function Key	Meaning

Application Logs	The *Log Selection* dialog box appears. You choose the processes for which you want to display the log. You can choose from the following processes: • Initial Filling • Roll Up • Compress InfoCube • Delete Request • Change Run • Check Choose ✔. The *Analyze Application Log* screen appears.
BIA Monitor	The *BI Accelerator Monitor Quick Overview* screen appears (see Using the BI Accelerator Monitor [Page 714]).
BIA Index Properties	If a BI accelerator index is available, the *Maintain BIA Index Properties* dialog box opens. You can specify the following settings: • Always store BIA index data completely in the main memory. This setting is advisable if enough main memory is available, you constantly require optimum response times, and the index is used frequently (see also Checking BI Accelerator Indexes (Transaction RSRV) [Page 723], test *Load BIA Index Data into Main Memory*). • Change the status of the BIA index: *Active* or *Inactive* (see scenario 2). Further information about the index is also provided: *Last Changed By*, *Date* and *Time* of last change, *Index Type* (see Technical Information About the BI Accelerator Engine [Page 702]).

Prerequisites

Communication between the BI system and the BI accelerator server takes place using RFC modules. To connect a BI accelerator server to the BI system, you must make the following settings:

- Set up the RFC destination for the BI accelerator server (transaction SM59). For more information, see Customizing under *SAP Customizing Implementation Guide → SAP NetWeaver → Business Intelligence → Connectivity of TREX → Creation of RFC Destination in BI System*.
- Specify the RFC destination for the BI accelerator server (transaction RSADMIN). The *RFC BI Accelerator* parameter has to correspond to the above RFC destination.

Note on System Landscape

Only one BI accelerator server can be used for each BI system. This is because the master data tables stored in the BI accelerator server can be used by multiple BI accelerator indexes. However, this does not work if the data is distributed across various BI accelerator servers.

If you want to run BI accelerators with productive and test systems in a system landscape, we recommend using a separate BI accelerator server for each BI system.

The InfoCube for which you want to create a BI accelerator is active and filled with data.

Process Flow

Access from Data Warehousing Workbench

1. You are in the Data Warehousing Workbench in the *Modeling* functional area. In the navigation window, choose *InfoProvider*. In the InfoProvider tree, navigate to the InfoCube with the queries you want to optimize using the BI accelerator index.
2. In the context menu of the InfoCube, choose *Maintain BI Accelerator Index*. The first dialog box for the BIA index maintenance wizard appears.

Access from Transaction RSDDV

1. On the *Aggregate/BI Accelerator Index: Select InfoCube* screen (transaction RSDDV), select the required InfoCube.
2. Choose *BIA Index*. The first dialog box for the BIA index maintenance wizard appears.

Scenario 1

You call the BIA index wizard for an InfoCube that does not yet have a BI accelerator index.

Step 1: Creating a BI accelerator index

When you execute this step, the system creates the indexes for the tables of the InfoCube star schema on the BI accelerator server, as long as they have not already been created by other BI accelerator indexes. These tables consist of the fact and dimension tables of the InfoCube as well as the master data tables that contain the required SIDs, the S, X, and Y tables of the InfoObjects. A "logical index" is also created. This contains the metadata of the BI accelerator. Finally, the system activates the BI accelerator index.

If the aggregate was filled successfully, the status in the *Object Version* column on the *Index Info* tab page switches to .

This step may take a few minutes if the individual tables are very large and have split indexes on the BI accelerator server. The more parts into which the index is being split, the longer the duration of the activation step. For more information about split indexes, see Technical Information About the BI Accelerator Engine [Page 702].

To use the BI accelerator index in reporting, you have to fill it with data. To schedule a background job to fill the BI accelerator index, choose *Continue*.

Step 2: Filling a BI accelerator index

The dialog box for specifying start date values appears. Specify when you want the fill job (RSDDTREX_AGGREGATES_FILL) to run in background processing and choose .

When you execute this step, the system starts a process in the background that reads the data in the tables of the InfoCube star schema from the database and writes them to the corresponding indexes on the BI accelerator server. If the index of a master data table (S/X/Y tables) has already been created and filled by another BI accelerator index, only those records that have been subsequently added have to be indexed (read mode/fill mode "D" during indexing).

If the aggregate was filled successfully, the status in the *Object Status* column on the *Index Info* tab page switches to .

Reading the data from the database and writing the data to the BI accelerator server can be performed in parallel in the BI system in different ways. To do this, maintain the system parameters in the BI accelerator monitor.

For more information about the steps for creating and filling a BI accelerator index, see Activating and Filling BI Accelerator Indexes [Page 708].

Step 3: Completing BI accelerator index maintenance

After the BI accelerator has been filled, you can choose *Cancel* to return to the source transaction or *Continue* to continue to the first part of BI accelerator index maintenance. The BI accelerator index is available and can be used for queries.

Scenario 2

You call the BIA index wizard for an InfoCube that has a BI accelerator index that is already filled with data.

Step 1: Deleting a BI accelerator index

Since an active and filled BI accelerator index that can be used for reporting is already available, you can either temporarily deactivate it or delete it at this point. This can be useful if you want to ensure, for performance purposes or analysis of data consistency, that the system is not using a BI accelerator index.

- To delete the BI accelerator index, choose *Continue*.

  The system deletes the definition and the settings of the BI accelerator index in the BI system and the logical index (metadata) and all indexes for the tables of the enhanced star schema of the InfoCube on the BI accelerator server. The only exceptions are the indexes for the master data tables that are still being used by other BI accelerator indexes.

- To deactivate the BI accelerator index temporarily, choose *BIA Index Properties*. The *BI Accelerator Index Properties* dialog box appears. Choose *Inactive* as the status of the BI accelerator index and choose .

  A BI accelerator that is switched off is **not** used when a query is executed. Since BI accelerator indexes that are switched off must also be consistent, you do not have to activate the BI accelerator index again or fill it when you switch it back on.

Scenario 3

You call the BIA index wizard for an InfoCube that already has an active BI accelerator index, but has not yet been filled or completely filled with data. The full process is either terminated or not even started.

Step 1: Deleting or continuing to fill a BI accelerator index

Since an active BI accelerator index that can be used for reporting is already available, you can either continue to fill it with data or delete it at this point. You can see the status of the individual indexes in the *Messages from Previous Step* area of the screen.

- To fill the BI accelerator index, choose *Continue Filling*.

- To delete the BI accelerator index, choose *Delete*.

4.4.6.3 Activation and Provision of Data

Use

The following sections explain how you activate BI accelerator indexes and fill them with data.

- If you have created a BI accelerator index, you have to activate it and fill it with initial data before you can use it when you execute a query. See Using the BIA Index Maintenance Wizard [Page 703]. For more information about the technical details, see Activating and Filling BI Accelerator Indexes [Page 708].
- If you have loaded new data packages (requests) into the InfoCube, you have to roll up these data packages into the BI accelerator index before the data is available in reporting. See Rolling Up Data in BI Accelerator Indexes [Page 710].

4.4.6.3.1 Activating and Filling BI Accelerator Indexes

Purpose

If you want to use a BI accelerator index with an InfoCube when you execute a query, you first have to create and activate a BI accelerator index and fill it with initial data. You use the BIA index maintenance wizard for this purpose (see Using the BIA Index Maintenance Wizard [Page 703]).

For more information about the global indexing parameters, see Global Parameters for Indexing [Page 721].

Additional technical information about these processes is provided in the following documentation.

Process Flow

Indexing Process by Table/Index on the BI Accelerator Server

The system performs the following steps in order to create an index on the BI accelerator server and make the data visible.

The name of the index is generated from the *System ID* and *Table Name*: <<system ID>>_<<table name>>. The system deletes the first forward slash from the table name and replaces the second with a colon.

- **Create**: For a table, the system creates the index on the BI accelerator server in accordance with the table properties. The system also determines how many parts the index is to be split into, depending on the present size of the table.
- **Index**: The data is transferred and written to a temporary file on the BI accelerator server.

- **Prepare optimize**: The data in the temporary file is formatted (compressed, coded and so on) as required for search and aggregation. Depending on how the index is distributed, this step can take longer than the indexing step.
- **Commit optimize**: The previously optimized data is made visible. If you perform rollback for an index, the system rolls back the data to the last commit optimize.

Example: Log messages for individual steps

	Index 'BR8_BI0:XCOORDER' created for BI accelerator index
	Index 'BR8_BI0:XCOORDER' filled for BI accelerator index (records written …)
	Prepare optimize for BI accelerator subindex 'BR8_BI0:XCOORDER'
	Commit optimize for BI accelerator subindex 'BR8_BI0:XCOORDER'

The logs for the initial fill/indexing of a BI accelerator index are in the application log under object “RSDDTREX“, subobject “TAGGRFILL“.

Competing Processes During Indexing

You can activate and fill BI accelerator indexes for different InfoCubes simultaneously.

However, overlaps may occur if several indexing jobs try to index the same master data tables simultaneously. In this case, the first job locks the table and performs indexing. The other jobs see the lock and schedule the indexing run to take place later. If no new data is loaded in the meantime, the system simply checks that indexing was performed successfully by the competing job. This step is necessary to avoid the system setting a BI accelerator index to “active” when the index is not actually available on the BI accelerator server because the job was terminated.

The subsequent jobs try a total of five times to start the indexing process or determine the status of the index. If this is not possible due to a long-running process or termination, the system terminates the entire indexing process for the BI accelerator index and notes the InfoCube affected by the lock process. You have to wait until the current program has finished or the error has been fixed before restarting the indexing process.

Example: Log for initial indexing with competing processes

	Load to index for table '/BI0/SVC_PAYM2' locked by competing job
	InfoCube of competing process: 'ZBWVC_003'
	Lock for table '/BI0/SVC_PAYM2'. Job will be restarted later
	...
	No new data for index of table '/BI0/SVC_PAYM2'
	BI accelerator index for InfoCube '0BWVC_003' filled successfully

Ability to Restart Processes

If a process is terminated by the user or the system during the initial data fill, you can restart the process by choosing the *Continue Filling* option in the corresponding dialog box of the BIA index maintenance wizard (see Using the BIA Index Maintenance Wizard [Page 703], scenario 3).

If indexing is terminated when the *Commit Optimize* is called, this is more problematic. After the *Commit Optimize* has been called, the data is visible and you can no longer perform rollback. This process is normally very quick and has an extremely low error rate. However, if indexing is terminated at the point at which the *Commit Optimize* is called, the status of the

index is “unknown” to the system since the system does not know whether data is already visible. The system does not know whether the termination occurred after or just before the commit optimize ran on the BI accelerator server. To clarify the status of the index, you have to analyze the termination message and the status of the index carefully; you cannot automate this analysis. Therefore, when restarting the process, the system normally re-indexes indexes that have status ”unknown” and notes records them in the log. The fact index is the exception: Since this is usually the largest BI accelerator index and re-indexing it takes a long time, the system does not automatically re-index the fact index. Instead the system terminates the process and gives the index status “unknown“. In cases like this, it may be useful to analyze the log message and repair the index manually.

4.4.6.3.2 olling Up Data in BI Accelerator Indexes

Use

As with relational aggregates, the data consistency of the InfoCube and BI accelerator index is based on request handling in the BI system. When you load new data packages (requests) into the InfoCube, these are not immediately available for use in a BI accelerator index for reporting purposes. As with aggregates, the process that writes new data to the BI accelerator index is **rollup**.

Integration

- If you replace the relational aggregates in an InfoCube with a BI accelerator index, you do **not** have to make further changes in the process chains or other settings. The process and the associated programs are identical.

The compression of data packages after rollup, as performed with aggregates to improve efficiency (see Efficiently Loading Data to Aggregates [Page 691] in section *Setting Automatic Compression*), does not apply to BI accelerator indexes because the data on the BI accelerator server already exists in a read-optimized format. However, it is useful to rebuild the BI accelerator index if the InfoCube is compressed heavily after rollup (see System Response Upon Changes to Data: BI Accelerator Index [Page 711] in section *Compression*).

- You can use delta indexes to speed up the rollup process. For information about optimizing the performance of BI accelerator indexes that are used particularly frequently, see Improving Efficiency Using BI Accelerator Delta Indexes [Page 713].

Prerequisites

- New data packages (requests) have been loaded into an InfoCube.
- BI accelerator indexes for this InfoCube have been activated and filled with data.

Features

When you rollup data for an InfoCube, the system first loads the new data into any aggregates that exist in the InfoCube, and then determines the delta of the missing records for all the tables that have an index in the BI accelerator index of the InfoCube and indexes it. If new SIDs are generated when transaction data is loaded, the system also writes new records to the indexes of the S, X and Y tables. When the system has indexed all the indexes successfully, the data of the most recent request is released for reporting.

Activities

As with relational aggregates, you only have to exit data rollup after loading transaction data.

For more information about the different execution modes for this activity, in particular the recommended execution types *Including Rollup of Data Packages As a Process in a Process Chain* and *Starting Rollup of Data Packages Manually*, see Rolling Up Data in Aggregates [Page 687]

In InfoCube administration, where you can see whether a rollup is missing, running or successful, the system does not differentiate between whether the InfoCube has aggregates or a BI accelerator index.

4.4.6.4 System Response Upon Changes to Data

Use

The following sections outline the points that you have to take into consideration if the data of the underlying InfoCube changes.

- For information about hierarchy/attribute change runs with BI accelerator indexes, as well as special cases where it is useful to restructure a BI accelerator index, see System Response Upon Changes to Data: BI Accelerator Index [Page 711].
- For information about further optimizing the performance of particularly frequently used BI accelerator indexes, see Improving Efficiency Using BI Accelerator Delta Indexes [Page 713].

4.4.6.4.1 System Response Upon Changes to Data: BI Accelerator Index

Use

Since, like aggregates, BI accelerator indexes are affected by changes to master data, they are also affected by hierarchy/attribute change runs.

If an InfoCube that forms the basis of a BI accelerator index is later compressed or data is deleted from it, we recommend that you rebuild the BI accelerator index.

Features

Hierarchy/Attribute Change Run

Since the data in **master data tables** (X and Y tables) is stored in indexes on the BI accelerator server, BI accelerator indexes, like aggregates, are affected by changes to master data. However, in contrast to aggregates, the fact tables do **not** contain the current data for the master data. Therefore, you do not have to run the potentially time-consuming delta calculations that you have to run for aggregates. Instead, you only transfer the changed records from the master data tables and change them in the indexes on the BI accelerator server. In most cases, this is considerably quicker than modifying aggregates.

Since the **hierarchy tables** are **not** in the BI accelerator index either, there is **no** pre aggregation on specific hierarchy levels, as is the case with aggregates. Again, calculation and modification is unnecessary. However, as with the BI hierarchy buffer, some views of hierarchies that occur in queries are stored on the BI accelerator server as temporary indexes so that they can be reused. If the hierarchy is changed, these temporary indexes have to be deleted.

The system changes both the master data and the temporary hierarchy indexes during the hierarchy/attribute change run. In this process, the aggregates and BI accelerator indexes for the relevant objects are determined for the previously changed InfoObjects that are selected. As before, the system first modifies the aggregates in accordance with the changes (see System Response Upon Changes to Master Data and Hierarchies [Page 689]) and then runs the two quick processes described for the relevant BI accelerator indexes:

- The X and Y indexes are filled with the changed records.
- The hierarchy buffer is deleted from the BI accelerator index.

Finally, the system activates the master data and displays the changed aggregates and BI accelerator indexes with the new data for reporting.

Compression

With BI accelerator indexes you do not have to compress after rolling up data packages. The data on the BI accelerator server already exists in a read-optimized format.

However, in the following cases it may be useful to rebuild the BI accelerator index, although this is not strictly necessary.

A BI accelerator index is created for an InfoCube that is not aggregated, or a large number of data packages are later loaded to this InfoCube. If you compress this InfoCube, more data is contained in the BI accelerator index than in the InfoCube itself and the data in the BI accelerator index is more granular. If compression results in a large aggregation factor (>1.5), it may be useful to rebuild the BI accelerator index. This ensures that the dataset is reduced in the BI accelerator index too.

Non-cumulative InfoCubes, that is InfoCubes with at least one non-cumulative key figure, should still be reconstructed in large intervals after compression. We recommend this especially if the time to calculate the markers at query runtime is large.

Deleting Data

If you delete data from the InfoCube selectively, the BI accelerator index has to be rebuilt. When you execute selective deletion, the system automatically deletes the affected BI accelerator index.

When you delete a data package (that is not aggregated) from an InfoCube, the index for the package dimension table is deleted and rebuilt. The facts in the fact index remain but are “hidden“ because they are no longer referenced by an entry in the package dimension table. Therefore, more entries exist in the index than in the table of the InfoCube. If you regularly delete data packages, the number of unused records increases, increasing memory consumption. This can have a negative affect on performance. In this case you should consider rebuilding the BI accelerator index regularly.

4.4.6.4.2 Improving Efficiency Using BI Accelerator Delta Indexes

Use

You can create a delta index for a BI accelerator index. If a delta index exists, the system does **not** write to the main index during each delta indexing or each indexing activity (except the initial filling/indexing) and the main index is not optimized. Instead, the system writes data to a second index which has the same structure as the main index but is usually smaller. The smaller the delta index, the faster the subsequent optimize procedure and therefore, the whole process of rolling up data or making modifications after a hierarchy or attribute change run.

As read performance deteriorates the larger the delta index gets, we recommend that you only switch on the delta index for essential indexes such as fact indexes and X/Y indexes. This improves performance when you modify data after a hierarchy or attribute change run.

Integration

We recommend that you regularly merge the delta indexes with your main index so that read performance is not negatively affected. You can do this in several ways:

- On the *Analysis and Repair of BI Objects* screen (transaction RSRV), area *BI Accelerator* → *BI Accelerator Performance*, you can select the *Size of Delta Index* elementary test. You can choose *Correct Error* to access repair mode and then execute a MERGE for the indexes. For more information about analyzing BI accelerator indexes in the analysis and repair environment, see Checking BI Accelerator Indexes (Transaction RSRV) [Page 723].
- You can schedule program RSDDTREX_DELTAINDEX_MERGE.

Activities

To set the delta index for a BI accelerator index, on the *BI Accelerator Monitor* screen (see Using the BI Accelerator Monitor [Page 714]), choose *BI Accelerator* → *Index Information* → *Set Delta Index.* The *Delta Index Properties* dialog box appears.

Switching On the Delta Index

In the *Delta Index* column, set the corresponding indicator if you want the table to use a delta index.

The new setting takes effect with the next delta indexing operation.

Switching Off the Delta Index

You reset the setting for the delta index in the same way.

Before the next indexing operation, the system merges the delta index and the main index. If the delta index is already very large, the next process may take longer.

4.4.6.5 Status Display and Check Tools

Use

The following sections outline how you get information about the structure and status of BI accelerator indexes and list the checks that you can perform.

- If you want an overview of the technical status of the BI accelerator and access to actions to restore the status, see Using the BI Accelerator Monitor [Page 714] for the relevant information.
- For more information about checking that the data records of a BI accelerator index are correct, checking the performance of BI accelerator indexes, or specifically rebuilding one or all BI accelerator indexes, see Checking BI Accelerator Indexes (Transaction RSRV) [Page 723].
- For an overview of the runtimes of specific subprocesses in BI accelerator index maintenance, see Statistics for Maintenance Processes of BI Accelerator Indexes [Page 732].
- You have various options for recording traces. For more information, see Traces for BI Accelerator [Page 733].
- If due to a communication problem you do not want the BI accelerator server to be available, you can switch on a fallback solution. To do this, you have to enter at least one e-mail address in table RSDDTREXEMAIL. The BI system saves the time of the error in table RSDDTREXHPAFAIL and sends the relevant information to the e-mail address entered in table RSDDTREXEMAIL.

  The recipient of the e-mail should trigger measures to remove the cause of the problem and delete the entry in table RSDDTREXHPAFAIL. When the entry has been deleted, queries automatically read from the BI accelerator server again and not from the database.

  For 30 minutes after the BI accelerator server goes down, or until the entry in table RSDDTREXHPAFAIL has been deleted, the system directs all query requests for an InfoCube with a BI accelerator index to the database. The system tries to use aggregates for this InfoCube, if they are available.

  After 30 minutes, if the problem is not resolved or if the time stamp entry is not deleted or changed, the system directs query requests to the BI accelerator server again. If the problem still exists, the system writes a new time stamp and redirects queries to the database again for the next 30 minutes.

  As soon as the BI accelerator is available, the system automatically sends all queries to the BI accelerator index of the affected InfoCube.

4.4.6.5.1 Using the BI Accelerator Monitor

Use

The BI accelerator monitor is used for the technical administration and maintenance of the BI accelerator. It provides an overview of the current status of the BI accelerator.

The BI accelerator monitor displays the results of the consistency checks. These checks run periodically on the BI accelerator. If problems occur, the system automatically proposes appropriate actions. These actions involve BI accelerator repair functions which can be used to fix problems.

To estimate the status of the BI accelerator quickly, there is a **quick overview function** for the BI accelerator monitor that summarizes the most important results.

The detailed BI accelerator monitor offers a detailed, technical overview of the hardware, BI accelerator services, any trace files that exist and the BI accelerator indexes.

In the BI accelerator monitor, it is **not** possible to maintain the indexes on the logical level of the BI accelerator InfoProvider. Use BI accelerator index maintenance instead (see Using the BIA Index Maintenance Wizard [Page 703]).

Queries work with the accelerator component of the BI accelerator, and the BI accelerator monitor works with the alert server. The accelerator components and the alert server are independent services within the BI accelerator. For this reason, queries can run without errors while at the same time, the BI accelerator monitor displays an error in the alert server.

Integration

Access from Data Warehousing Workbench

You are in the *Administration* functional area of the Data Warehousing Workbench. In the navigation pane, choose *Monitors* → *BI Accelerator Monitor*. The quick overview of the BI accelerator monitor appears.

Access from Transaction RSDDV

You have called the BIA index maintenance wizard for the InfoCube. In the dialog box for the BIA index maintenance wizard, choose *BIA Monitor*. The quick overview of the BI accelerator monitor appears.

Integration in the CCMS Monitoring Architecture

If you do not only want to monitor your BI accelerator locally using the BI accelerator monitor, but want to monitor your BI system **centrally** with the BI accelerator alongside other systems, you can use the central alert monitor (CCMS monitoring, transaction RZ20).

The information and the results that you can find in the *BIA Check Results* screen area of the BI accelerator monitor in the *Check Results* area on the *Summary* and *Current Results* tab pages are displayed.

You can find more information about the connection of the BI accelerator monitor to the CCMS monitoring framework in SAP Note 970771.

More information about the CCMS monitoring framework: Alert Monitor [External]

Prerequisites

The BI accelerator is based on TREX technology. For more information, see Performance Optimization with BI Accelerator [Page 700].

Features

BIA Results of Check Screen Area

***Test Results* area**

Here the system displays the results of the BI accelerator consistency checks.

Tab Page	Description
Summary	The start view shows a summary of the current results of the check. The system tries to summarize the current results of the check in such a way that the status of the BI accelerator can be repaired in as few actions as possible. The status display shows: if the status is ok if there are messages or warnings if there are errors for the status With status and , the system usually displays an action that can fix the problem. If the proposed action can be started from the BI system, you initiate this action by choosing in the *Execute* column. For each action, the user can display an explanatory long text by choosing (*Display Long Text*). If details are available for a check, you can call them by choosing (*Details Available*) in the *Details* column. The system displays the details in the *Check Details* screen area.
Current Results	This view displays the current results of the consistency checks that were performed. The status of the checks are indicated by the colors , or . The system displays general information about each check: the length of the check (in seconds), the date and time at which the check was started, and the check set within which the check was started. For each check, the user can display an explanatory long text in the *Long Text* column by choosing (*Display Long Text*). If a check determines that an action is required to solve the problem, the system displays the action in the table. You start the action by choosing in the *Execute* column. Choose to display an explanatory long text. If details are available for a check, you can call them by choosing (*Details Available*) in the *Details* column. The system displays the details in the *Check Details* screen area.

History	This view returns the results of previous runs for the BI accelerator consistency checks so that you can track developments or changes in the results. The system displays general information about each check: the length of the check (in seconds), the date and time at which the check was started, and the check set within which the check was started. For each check, the user can display an explanatory long text in the *Long Text* column by choosing (*Display Long Text*). Since this information is not current, the system does not propose actions.

You can find out about the state of the BI accelerator at a scheduled moment in time by e-mail. More information: section *Menu BIA Checks*.

***Check Details* area**

If more details are available for the results of the check, the user can display them here.

If problems are listed with BI accelerator servers, you can display the affected servers in the details area.

BIA Actions Screen Area

***Execute Actions* area**

In the *Execute Actions* screen area, the user can directly execute the most important actions to resolve BI accelerator problems. The BI accelerator differentiates between actions that the user can start from the BI system and those that the user must carry out in the BI accelerator.

Example of an action that the user must carry out in the BI accelerator: Actions for the trace files of the BI accelerator.

On the *Current Results* tab page in the *Check Results* screen area, the BI accelerator proposes actions for check results that have status or . If these are actions that can be executed from the BI system, you can execute them directly by choosing .

In the *Execute Actions* screen area, the BI accelerator monitor collects all the proposed actions. It sets the indicator if the action can be started from the BI system. A *Proposal* field is displayed alongside the proposed actions.

Here the system supports the direct execution of the following actions:

Action	Description
Restart host	This action restarts the BI accelerator hardware.
Restart BIA server	This action restarts all the BI accelerator servers and services. This includes the name server and index server.
Restart BIA index server	This action only restarts the index server. (The name servers are **not** restarted.)
Rebuild BIA indexes	If a check discovers inconsistencies in the indexes, you can use this action to delete and rebuild all the BI accelerator indexes.

Reorganize BIA landscape	If the BI accelerator server landscape is unevenly distributed, this action redistributes the loaded indexes on the BI accelerator servers.

The actions *Restart Host, Restart BIA Server*, and *Restart BIA Index Server* are hierarchically related: If the host is restarted, the server is automatically restarted so that this action no longer has to be started explicitly. For example, the *Restart BIA Server* action includes a restart of the BIA index server. Therefore, as soon as a higher-level option is selected, the system automatically sets the indicator for the lower-level selection boxes and deactivates them for the selection.

***BIA Action Messages* area**

The log display in the *BIA Action Messages* screen area shows information about the processes in the BI accelerator monitor.

If the system reads status information (gets check results), it writes this to the log, for example: *Status Information Read from BIA*.

Each message has a status (, ,). Where appropriate, you can also display the explanatory long text by choosing .

The *Detail Level* is usually rule 1, unless the messages are structured hierarchically.

You use (*Refresh Messages*) to refresh the log display.

You use (*Delete Messages*) to delete the messages in the log.

Toolbar Functions

The following functions are available in the toolbar:

Function	Description
(Refresh *BIA Log and Actions*)	Refreshes the monitor. The system displays the current results of the BI accelerator checks again and makes new proposals for actions.
BIA Availability	Uses an RFC availability test to check the availability of the connection to the BI accelerator. If **no** connection to the BI accelerator is available, necessary measures are initiated.

Switch On BIA Load Monitor or *Switch Off BIA-Load Monitor*	Calls the BI accelerator load monitor in a separate window that refreshes itself independently. In this window, you can see the following BI accelerator key figures: *Host:Port*: Host and port of the BI accelerator *Memory Process*: Memory usage of TREX server process *Total Memory*: Memory usage of all processes *Memory Available*: Available memory *CPU of All Processes*: CPU usage of all processes *CPU Process*: CPU usage of TREX server processes *Response Time*: Average response time of the last queries *Queries*: Queries per second *Requests*: Number of external requests *Requests Including Internal*: Number of external and internal requests *Requests Active*: Number of active requests *Hanging Requests*: Number of hanging requests You can only start **one** load monitor. Since the load monitor is started in a new window, it uses a new mode. Make sure that a mode is available before you activate the load monitor. For technical reasons, the load monitor window is kept open. If it is hidden by other windows, you can access it using the key combination ALT+TAB. You can continue to work in this window or close it. You can only end the load monitor from the BI accelerator monitor: As soon as you start the load monitor, the pushbutton function changes to *Switch Off BIA Load Monitor*. After you have stopped the load monitor, the system resets the mode to the start view.

Menu BIA Checks

You can broadcast the results of the consistency checks of the BI accelerator (an overview of the summary and the current results) by e-mail from the entry *Schedule E-Mail Notification*.

In the *Start Time* dialog box, you can schedule the respective job for the required point in time.

The e-mail is then created and broadcast to all addresses in the RSDDTREXEMAIL table.

Menu *Goto*

You can choose the following options from the *Goto* menu:

Menu Option	Meaning

Analysis of BI Objects	The *Analysis and Repair of BI Objects* screen appears. More information: Checking BI Accelerator Indexes (Transaction RSRV) [Page 723]
BIA Index Maintenance	The *Maintaining Aggregates/BIA Index: Select InfoCube* screen appears (transaction RSDDV). More information: Using the BIA Index Maintenance Wizard [Page 703].
TREX Administration Tool	The *TREX Administration Tool* screen appears.

Menu *BI Accelerator*

You can choose the following options from the *BI Accelerator* menu:

Menu Option	Meaning
Execute Action	You start required actions directly from the BI system (see **BIA Actions screen area** above).
Switch On BIA Load Monitor or *Switch Off BIA Load Monitor*	You can call the load monitor of the BI accelerator in a separate window (see **Toolbar Functions** above).
Index Checks	With the *Execute/Display Index Checks* menu option, the system executes the following checks once a day (always at 0:00:01) and then displays the results: • *Check size of delta indexes for BIA indexes* • *Compare size of InfoCube fact table with fact index* • *Find indexes with status "unknown"* • *Check: Table index relation* These checks are quick to run and provide important information about performance and consistency. They can be found in the analysis and repair environment (see Checking BI Accelerator Indexes (Transaction RSRV) [Page 723]). If you do **not** want to run these checks regularly, you can choose the *Schedule Index Checks* option from the menu and specify that the system should not schedule the checks.
Index Information	You can call the following functions from this menu option: • *Change Global Parameters*: See Global Parameters for Indexing [Page 721]. • *Display All BIA Indexes*: See BI Accelerator Index Design [Page 722]. • *Set Delta Index:* See Improving Efficiency Using BI Accelerator Delta Indexes [Page 713]. • *Switch BIA Indexes for Queries On/Off*

Activities

You get detailed information about the status of the BI accelerator and the check results.

The system proposes actions to correct errors in the BI accelerator, if applicable.

If these actions can be started in the BI system, you can trigger them there immediately.

4.4.6.5.1.1 Global Parameters for Indexing

Use

You use this dialog box to edit the default values for the global indexing parameters.

Integration

To navigate to this dialog box, on the *BI Accelerator Monitor* screen (transaction RSDDBIAMON), choose *BI Accelerator* → *Index Information* → *Change Global Parameters*.

Features

Global Indexing Parameters

Name	Description	Value (Changeable)
BATCHPARA	*Number of Processes for Background Parallel Processing of Initial Indexing* Number of parallel background processes for initial indexing: F and E fact tables of the InfoCube are split into a number of blocks. The BACTHPARA parameter specifies the number of blocks. (The E table is split using the partitioning characteristic or another time characteristic; the F table is split using the request.) These blocks are then read from the database and written to the BI accelerator separately. (Note that a number of dialog processes, as specified in the NUMPROC parameter, are used for each background process.) This applies to initial indexing only and **not** roll up. You can change the values for these parameters in transaction RSBATCH (process type TREXINDEXP).	03
NUMPROC	*Number of Processes for Parallel Processing Using aRFC Dialog Processes During Indexing* The data is read from the database table on a package-by-package basis using **Open Cursor** and **Fetch**. The system calculates the package size from the width of the table and the default value for the package size (in bytes) in the system. If you set a degree of parallelization that is greater than one, each of these packages is indexed in a new asynchronous job in dialog mode. When the indexing job for the package is started, the system reads a new package immediately. Ideally, you should reduce the indexing time to the time that is required to read the data and pack it for the RFC module. If the degree of parallelization is equal to one, the system performs serial processing. The optimization of the BI accelerator index cannot be parallelized on the BI side. However, optimization automatically runs in parallel on the BI accelerator server if the index is split.	5
PKGSIZE	*Package Size in Bytes for Internal Tables During Indexing Using aRFCs*	10.000.000

SUBPKGSIZE	*Package Size (Rows) for **Export to Buffer** During Indexing Using aRFCs* During dialog parallel processing with the NUMPROC parameter, packages with size PKGSIZE are read and indexed using aRFCs. To do this, the data packages must be transferred using *Export to Data Buffer*. This is done on a package-by-package basis since it requires a large amount of CPU and memory resources. The SUBPKGSIZE parameter specifies the number of rows for the package. We recommend a value between 10,000 and 20,000.	20.000

Activities

You use this dialog box to edit the values of the global indexing parameters.

4.4.6.5.1.2 BI Accelerator Index Design

Use

BI accelerator design provides information about the structure, properties and status of the BI accelerator index and its tables/indexes.

Tables that are part of the enhanced star schema of the selected InfoCube and are required in the corresponding BI accelerator index form part of the description of the BI accelerator index.

All the dimension tables of the InfoCube are required for the star schema of the BI accelerator index. The E and F fact tables of the InfoCube form one fact index. Of the master data tables, only the X and Y tables (which contain the SIDs) are required; the P and Q tables (which contain the key values) are not required. The SID tables (S tables) are required if the InfoObject has a non-numeric key.

Integration

BIA index-specific information can be displayed in BIA index maintenance and in the BI accelerator monitor.

BIA index maintenance

As soon as a BI accelerator index has been created, the system displays information about its tables and indexes on the *Index Info* tab page (see Using the BIA Index Maintenance Wizard [Page 703]).

BI accelerator monitor

If you choose *BI Accelerator* → *Index Information* → *Display All BIA Indexes*, the *Information about BIA Indexes* dialog box appears. The system displays all the BI accelerator indexes that exist in the system.

Features

In the BI accelerator monitor, the system shows more information than in BIA index maintenance. The following table provides an overview of this information. * indicates that the column is displayed in BIA index maintenance as well as in the BI accelerator monitor.

Description of a BI accelerator index

Column	Description
InfoCube	Technical name of the InfoCubes for which BI accelerator indexes have been created
Object Version	Status display: BI accelerator index is active. BI accelerator index is not active. See Activating and Filling BI Accelerator Indexes [Page 708]
Object Status	Status display: BI accelerator index is filled. BI accelerator index is not filled. See Activating and Filling BI Accelerator Indexes [Page 708]
Table Name *	Technical name of the relevant index on the BI accelerator server.
Table Size *	Specifies the approximate current size of the individual tables (number of data records), as calculated from the database statistics.
Index Status *	Status of index
with Delta Index *	Indicates that a delta index is being used for the BI accelerator index (see Improving Efficiency Using BI Accelerator Delta Indexes [Page 713]).
Multiple Usage *	With S, X and Y tables, this indicates that one of the tables is already being used by another BI accelerator index and therefore already exists as an index.
Last Changed By *	Name of user who made the last change.
Last Changed/Time Stamp *	Date and time of last change.

4.4.6.5.2 Checking BI Accelerator Indexes (Transaction RSRV)

Use

On the *Analysis and Repair of BI Objects* screen (transaction RSRV), various checks are available for:

- Testing for inconsistencies between the data in the InfoCube on the database and the data in the BI accelerator index (tests in the *BI Accelerator Consistency Checks* area)

- Testing whether a BI accelerator index is running with the most optimal performance (tests in the *BI Accelerator Performance Checks* area)
- Completely or partially building or rebuilding all BI accelerator indexes or a specific BI accelerator index (tests in the *BI Accelerator Repair Programs* area)

The exactness and duration of each of these checks vary.

For more information about building and using the analysis and repair environment, see Analysis and Repair Environment [Page 792].

Integration

In the *BI Accelerator Monitor*, you can specify that the system is to run a small number of tests on a daily basis. You do this by choosing *BI Accelerator* → *Execute/Display Index Checks*. For more information, see Using the BI Accelerator Monitor [Page 714].

Prerequisites

- The BI accelerator index you want to check has been activated and filled with data.
- Some tests work with statistics data (see tests: *Propose Delta Index for Indexes, Compare Size of Fact Tables with Fact Index*).

  As a prerequisite, the statistics have to be switched on for the relevant InfoProvider. You make this setting in statistics properties maintenance screen (on the *Data Warehousing Workbench* screen, choose *Tools* → *Settings for BI Statistics*). For more information, see Statistics for Maintenance Processes of BI Accelerator Indexes [Page 732].

Features

The following tests are available under *All Elementary Tests* → *BI Accelerator*:

BI Accelerator Consistency Checks

Master Data and Transaction Data

Compare Data in BI Tables and BIA Indexes (*Check Table Index Content*)

The system compares the content of each individual table with the content of the corresponding index on a record-by-record basis. This check is only suitable for tables or indexes that do not contain a large amount of data, such as dimension tables, certain SID tables (S) and attribute tables (X and Y). This is not generally the case with fact tables. If a table contains 10,000 records or more, it is **not** checked.

In some situations, the content of the indexes of the BIA index may differ from the content of the corresponding database table. This may be the case if requests have been deleted from the InfoCube or if an InfoCube has been compressed.

Check Sums of Key Figures of BIA Queries (*Check Key Figure Sums Internally*)

First the system executes a query on the BI accelerator index, which is aggregated using all key figures. Next, all the characteristics and navigation attributes that exist in the InfoCube are included in the drilldown individually and the totals are calculated. The system compares the result with the result of the first query. This test checks the completeness of the **join** path from the SID table, through the dimension table, to the fact tables.

Runtime: Depends on the number of characteristics and navigation attributes and on the number of records in the fact table.

If the test shows that the data is incorrect, you have to rebuild the BIA index and the indexes for the master data tables.

Check Sums of Key Figures of BIA Queries with Database (*Check Table Index of Key Figure Totals*)

Similar to mode *Internally Check Key Figure Totals*, the system executes highly-aggregated queries and compares the results of the InfoCube in the database with those of the BI accelerator index.

For large InfoCubes the runtime may already be considerable, since queries to the database take longer.

Check Existence of Indexes for Database Tables (*Table-Index Relation*)

An index is created for almost every table of the BI InfoCube enhanced star schema: fact (F) tables, dimension (D) tables, SID (S) tables and attribute tables (X and Y); the only exception is SID tables with numeric characteristic values.

This test checks whether the named indexes have been created on the BI accelerator server.

Runtime: Very fast

If the test reveals that an index is missing, rebuild the index for the table.

Check for Consistency Using Random Queries

The system creates random queries without persisting them. These random queries are only used for this test: The system reads the data once from the database and once from the BI accelerator. It then compares the results. If the results differ, an error message is output.

Note that there can be different results if the data of the InfoCube is changed between execution of the query on the database and in the BI accelerator (for example by a change run or by rolling up new requests).

You can verify the results by executing the program RSDRT_INFOPROV_RANDOM_QUERIES with the following parameters:

- InfoProvider: Name of the InfoCube
- Number of queries: 10
- Starting value of random generator
- Trace comparison: 'X'

You can leave all other values unchanged. The program can also be executed in the background and the results viewed in the spool list.

If you use the same starting value, the same random queries are generated; you can thus repeat the test.

Automatic repair is not available. If necessary, you must rebuild the BI accelerator index.

Verification of the Buffer Entries of the BIA Hierarchy Buffer

When queries in hierarchies are executed, the relevant hierarchy nodes are expanded to the relevant leaves. This leaf-node relation is saved in a temporary index in the BI

accelerator. The hierarchy buffer manages expanded hierarchies according to an LRU (least recently used) algorithm.

The check verifies whether all temporary indexes in the hierarchy buffer contain the correct data.

If the hierarchy buffer contains incorrect entries, write a customer message. This state is incorrect. If you urgently need to resolve the error, you can delete the entire hierarchy buffer. In this case, however, SAP will not be able to find the error.

Metadata

Check Definition of Logical Index

The system compares the definitions of each of the indexes for a BIA index with the current versions of the database tables. It checks whether the number, name, and type of the table fields in the database match the definition for the index on the BI accelerator server.

An index may have changed if, for example, the InfoCube was changed. If this is the case, the BI accelerator index has to be repaired (see test *BIA Index Adjustments After InfoCube Activation*).

Note that if you do not specify an InfoCube, the system executes the test for all InfoCubes that have a BI accelerator index.

If an index has been changed, the system deletes the old index, creates a new index with the correct definition, and fills it. All BI accelerator indexes that use this index are set to "inactive"; they are not available for reporting purposes during this time.

Runtime: Depending on the size of the table, this process may take some time.

Compare Index Definition in BIA with Table on Database

The system checks the logical index of a BI accelerator index. The logical index contains the metadata of the BI accelerator index, such as the **join** conditions and the names of the fields.

The logical index may change if, for example, the InfoCube has been changed. If this is the case, the BI accelerator index has to be repaired (see test *BIA Index Adjustments After InfoCube Activation*).

Note that if you do not specify an InfoCube, the system executes the test for all InfoCubes that have a BI accelerator index.

If the logical index has been changed, the system deletes the old index and creates a new index with the correct definition. The system temporarily sets the BI accelerator index to "inactive"; it is not available for reporting purposes during this time.

Find indexes with status “unknown“

The system checks whether BI accelerator indexes contain indexes that have the status "unknown" (= U). This only occurs in exceptional cases when the **commit** call (**commit optimize**) terminates during indexing. Since in this case it is not clear whether the data from the preceding indexing call is available, the affected indexes are rebuilt in repair mode.

BI Accelerator Performance Checks

Size of Delta Index

If you have chosen delta mode for an index of a table, new data is not written to the main index but to the delta index. This can significantly improve performance during indexing. However, if the delta index is large, this can have a negative impact on performance when you execute queries. When the delta index reaches 10% of the main index, the system displays a warning.

The system performs a **merge** for the index in repair mode. The settings are retained.

Propose Delta Index for Indexes

It is useful to create a delta index for large indexes that are often updated with new data. New data is not written to the main index, but to the delta index. This can significantly improve the performance of indexing, since the system only performs the **optimize** step on the smaller set of data from the delta index. The data from the delta index is used at query runtime.

The system determines proposals from the statistics data: Proposals are those indexes that received new data more than 10 times during the last 10 days. A prerequisite for these proposals is that the statistics for the InfoCube are switched on.

Data in the main index and delta index should be **merged** at regular intervals (see test *Size of Delta Index*).

In repair mode, the system sets the *Has Delta Index* property for the proposed indexes. The delta index is created when the data is next loaded for this index.

Compare Size of Fact Tables with Fact Index

The system calculates the number of records in both fact tables (E and F tables) for the InfoCube and compares them with the number of records in the fact index of the BI accelerator index. If the number of records in the BI accelerator index is significantly greater than the number in the InfoCube (more than a 10% difference), you can improve query performance by rebuilding the BIA index.

The following circumstances can result in differences in the numbers of records:

- The InfoCube was compressed after the BI accelerator index was built. Since the BI accelerator index is not compressed, it may contain more records than the InfoCube.
- Requests were deleted from the InfoCube after the BI accelerator index was built. The requests are deleted from the BIA index in the package dimension only. The records in the fact index are therefore no longer referenced and no longer taken into account when the query is executed; however, they are not deleted.

  Note that the database statistics for calculating the size of the fact table must be up to date, since the test does not recount; it uses the database statistics from the tables.

Load BIA Index Data into Main Memory

You use this test to load all the data for a BI accelerator from the file server into the main memory if the data is not already in the main memory.

This action is useful if you want to ensure that queries executed in the corresponding InfoCube achieve optimal performance the first time they are executed and do not have to read data anew from the file server.

Data for an index is deleted from the main memory, for example, when new data is added to this index (during roll up or a change run). In BIA index maintenance (choose *BIA Index Properties*, see Using the BIA Index Maintenance Wizard [Page 703]), you can also adjust the

settings for the BI accelerator index such that data is loaded automatically to the main memory every time changes are made.

Note that if you do not specify an InfoCube, the system executes the test for all BI accelerator indexes that are active and filled.

Delete BIA Index Data from Main Memory

You use this test to delete all data for a BI accelerator index from the main memory.

Master data indexes that are still required by other InfoCubes are **not** deleted from the main memory. The data on the file server is **not** deleted – the BI accelerator index is still active.

This action is useful if there is little space in the main memory on the BIA server and you have data in the main memory that can be deleted. This is useful in the following cases:

- There is data in the main memory that is no longer used or is rarely used.
- There is data in the main memory that does place a high load on system performance when the query is executed initially (and when the file server is read in the main memory).

If you do not specify an InfoCube, the system runs the test for all BI accelerator indexes that are active and filled.

Estimate Runtime of Fact Table Indexing

The system estimates the time required to fill the fact index. It uses the current parameter values for background and dialog parallel processing. The time taken is calculated from the processes available and the estimated maximal throughput of data records in the database, the application server, and the BIA server.

The calculated duration is an estimation; the load on the system, the distribution of data across block criteria and deviations during processing can all affect the actual time taken.

Estimate Memory Consumption of Fact Table Index

The system estimates the size of the fact table index of a BI accelerator index. In doing so, the system analyzes the data in the fact table and provides a projection.

Note that if data distribution is poor, the actual memory consumption can deviate from the projected value. A more exact analysis would demand more time than that required to rebuild the index, since the number of different values in the fact table needs to be determined for each column (**count distinct**).

BI Accelerator Repair Programs

Delete and Rebuild All BIA Indexes

All BI accelerator indexes in the system are deleted. If you selected the *Execute* option, the indexes are then recreated and filled. This is sometimes required for a successful restart with consistent data if a critical error occurs.

BIA Index Adjustments After InfoCube Activation

If an InfoCube is changed as a result of the addition of a key figure, for example, the system does not automatically adjust the BI accelerator index, since the relevant process may take a long time and can even require a partial reindexing.

When you execute this test, information about any changes identified are written to the log. The system makes the required changes in repair mode.

We recommend that you run this repair job as a background job, if required.

Rebuild All Master Data Indexes of a BIA Index

All indexes for master data tables in a BI accelerator index are rebuilt. This includes indexes for SID tables and attribute tables (X and Y tables). When an entire BI accelerator index is rebuilt, these tables are not always rebuilt since they are also used by other BI accelerator indexes. If this results in data consistency issues, it may be necessary to rebuild the indexes for the master data tables.

In repair mode, the system first deletes the relevant indexes and then recreates them. All BI accelerator indexes that use these indexes are set to "inactive"; they are not available for reporting purposes during this time.

The following tests are available under *All Combined Tests* → *BI Accelerator*:

Consistency Checks (Detailed)

Consistency Checks (Fast)

Performance Tests

Execution Modes

Execution Mode	Description
Schedule	The dialog box for specifying start dates appears. Specify the time(s) for the execution. You can view the results of the check in the protocols in the application log.
Correct Error	In repair mode, the system performs certain repair tasks. (Repair tasks are not available for all tests).

Evaluating Logs

1. In the *Selection of Check Mode for BI Accelerator Index* dialog box, choose *Display Logs*. The *Analyze Application Log* screen appears for object RSDDTREX, subobject TAGGRCHECK.
2. Enter the required data to restrict the number of logs.
3. Choose *Execute*. The system displays the results of the check.

Activities

You select the test(s) and specify the mode of execution. You can view the results of the check in the protocols in the application log.

4.4.6.5.3 Checking BI Accelerator Indexes (Check Center)

Use

From the *SAP NetWeaver BI accelerator monitor*, choose *Goto* → *Consistency Checks* to display the *BI Accelerator Data Consistency Check Center* screen. On this screen, you can check the data on the BI accelerator server, schedule these checks, and view the logs of checks that have already run. You can combine certain checks to form check sets.

Procedure

Creating a New Check Set

1. Give the check set a *description.*
2. Specify the *InfoCubes* of the BIA index for which the check set is to be executed. Input help is available. Multiple selection is possible.
3. Specify the *maximum degree of parallelization* if necessary. The degree of parallelization is only taken into consideration during background processing. The system starts one process (dialog) for each InfoCube, with a maximum of n processes running simultaneously (n = parameter value).
4. If necessary, set the indicator for the *Deactivate BIA Index for Queries if Error Occurs* option. If you set this indicator, the BIA index is immediately set to 'inactive' (cannot be used for queries) as soon as the check set displays incorrect data in the BIA index. This prevents incorrect data being used for reporting in the BIA. Note, however, that a check can display incorrect data even though the data is correct, for example, because a load process (master data or transaction data) has changed the data at the same time.
5. If you want an *e-mail* to be sent if an error occurs (if incorrect data is displayed), enter the address of the recipient in the relevant field.
6. If the check set is to be *executed immediately after the rollup of new requests to an InfoCube*, set the relevant indicator. The check set is then still part of the process (this is relevant for integration into a process chain), but the lock on the process is no longer valid, so that other processes are not disturbed. The check set is not executed for all InfoCubes, but only for the InfoCube for which the data was rolled up.
7. If the check set is to be executed *immediately after the change run*, set the relevant indicator. As previously, the check set is still part of the process, but the lock on the process is no longer valid. The check set is only executed for the InfoCubes whose BIA index was adjusted in the change run.
8. Each tab page contains a test. You can find the description of the test under *Details of Check.* Select the checks that are suited to your check set by setting the corresponding indicator for *Execute Test.* Choose the *check-specific* option.

Overview of Consistency Checks

Tab Page	Test Description
Data Compar.	For more information, see Checking BI Accelerator Indexes (Transaction RSRV) [Page 723]: *BI Accelerator Consistency Checks → Compare Data in BI Tables and BIA Indexes (Check Table Index Content)*

Totals in BIA	For more information, see Checking BI Accelerator Indexes (Transaction RSRV) [Page 723]: *BI Accelerator Consistency Checks → Check Sums of Key Figures of BIA Queries (Check Key Figure Sums Internally)*
BIA and DB Totals	For more information, see Checking BI Accelerator Indexes (Transaction RSRV) [Page 723]: *BI Accelerator Consistency Checks → Check Sums of Key Figures of BIA Queries with Database (Check Table Index of Key Figure Totals)*
Random Queries	For more information, see Checking BI Accelerator Indexes (Transaction RSRV) [Page 723]: *BI Accelerator Consistency Checks → Check for Consistency Using Random Queries*
Index Exist.	For more information, see Checking BI Accelerator Indexes (Transaction RSRV) [Page 723]: *BI Accelerator Consistency Checks → Check Existence of Indexes for Database Tables (Table-Index Relation)*

9. Save the check set. A check set ID is allocated and displayed.

Displaying and Changing an Existing Check Set

- To display an existing check set, place the cursor in the *Check Set ID* field and select the required check set from the input help.
- Change the parameter values of the selected check set and save it again. A new check set ID is **not** allocated.

Executing a Check Set

- Select an existing check set or define a new one.
- The checks of the check set are executed in the dialog (and not in parallel) using *Execute.* The check set does not have to be saved for this. After the check is complete, you automatically branch to the display of the results in the application log.
- Choose *Schedule* to open the *Start Time* dialog box. Here you can schedule the check set once or periodically in the background. The check set must be saved for this. The name of the scheduled job is BW_TR_RSDDTREX_INDEX_CHECK.
- You can also use program RSDDTREX_INDEX_CHECK to execute a check set. To do this, you need the check set ID of the check set, or you can choose the required check set from the input help. You can also use this program to integrate a check set into process chains. Choose *Logs* to call the logs.

Deleting a Check Set

- Select an existing check set, choose *Delete*, and answer *Yes* to the confirmation prompt.

4.4.6.5.4 Statistics for Maintenance Processes of a BI Accelerator Index

Use

To get an overview of the runtimes of specific subprocesses in BI accelerator index maintenance, you can display the RSDDSTATTREX table.

For the following processes, the system writes the runtimes of specific subprocesses to this statistics table:

- Initial indexing (see Using the BIA Index Maintenance Wizard [Page 703])
- Rollup (see Rolling Up Data in BI Accelerator Indexes [Page 710])
- Modifications after change runs (see System Response Upon Changes to Data: BI Accelerator Index [Page 711])

Integration

Some BI accelerator tests in the analysis and repair environment work with statistics data (see Checking BI Accelerator Indexes (Transaction RSRV) [Page 723], tests: *Propose Delta Index for Indexes, Compare Size of Fact Tables with Fact Index*).

Prerequisites

The statistics have to be switched on for the relevant InfoProviders. You make this setting in statistics properties maintenance (on the *Data Warehousing Workbench* screen, choose *Tools* → *Settings for BI Statistics*). For more information, see Maintenance of Statistics Properties [External].

Features

The statistics table contains the following information for each table that is indexed:

RSDDSTATTREX

Column	Description
STATUID	Unique identification key
TABLNM	Table name
CHANGEMODE	Specifies whether the process is part of a BI accelerator rebuild ("N"), the rollup ("R") or a modification after a change run ("C").
FILLMODE	Fill Mode: Full ("F"), delta ("D") or change run ("C")
TIMEACTIVATE	Time of activation
TIMEREAD	Time required to read data from the database
TIMEFILL	Time required for packing and indexing
TIMEOPTIMIZE	Time for **Prepare Optimize**
TIMECOMMIT	Time for **Commit Optimize**
REC_INSERTED	Number of indexed records

TSTPNM	User
TIMESTMP	Start time stamp

4.4.6.5.5 Traces for BI Accelerator

Use

When errors occur it can be useful to record system responses in the form of traces. SAP support has tools to evaluate these traces.

This may be the case, for example, if you are obtaining different results for a BI query depending on whether you use the BI accelerator.

Integration

To record traces for query execution, use the query monitor (see Query Monitor [External]).

To record performance traces, use the BI accelerator monitor (see Using the BI Accelerator Monitor [Page 714]).

Activities

Trace for Queries in the Query Monitor

Activation

In the query monitor you can execute and debug queries.

1. Select the query for which you want to record a trace.
2. Choose *Execute and Debug*. The *Debug Options* dialog box appears. The options are ordered in a hierarchy.
3. Choose *BIA Server* → *BIA Default Trace*.

   If you set the indicator for the *BIA Default Trace*, the system automatically activates all the traces listed under this option that log information about the query that is currently being executed.

   You can also choose a single trace type.

Overview of BIA default traces

Trace Type	Description
BIA Python Trace	The BI accelerator index server is traced. The system generates a Python program that can be executed. To find out the selections for a query, for example, support can reproduce a query (without recording the ABAP read interface).

BIA Plan Trace	The executor (a component of the BI accelerator) is traced. The system generates a program that can be executed. To analyze the steps that the BI accelerator executes (the execution plan), support can reproduce a query (without recording the ABAP read interface).
BIA ABAP Trace	The system records the parameterization of the read interface. To analyze the problems with the RFC server, for example, support can reproduce a query on the basis of the BI accelerator indexes (without the InfoCubes).
BIA Standard Trace	The system records the trace with particular internal settings (trace levels). The result is returned in the form of a text file, is linked to the query, and is only valid for this query. This trace records error messages. If, for example, a query throws an exception, you can replay the trace to receive more precise error messages.

Display

If you have activated one of the three trace types, the system displays the trace after the query has been executed. You can edit the trace file and save it locally.

Runtime problems may arise for large trace files. For this reason, you can also save the trace file without displaying it and editing it.

Performance Trace in the BI Accelerator Monitor

Activation

You can activate a performance trace in the BI accelerator monitor. This logs system responses. SAP support has tools for evaluating these system responses. The trace is written in save-optimized format (*.tpt).

To activate a trace, choose *Performance Trace* → *Start Trace Recording* from the menu.

A dialog box appears in which you set:

- whether you want to start the trace for a particular user
- when you want to stop the trace

For performance reasons, we recommend that you do not choose a time that is too far in the future.

In the status bar, the system shows how long trace recording has left to run (for example,

BI Accelerator Monitor (Trace Recording Still Active 00:10:30).

If a trace recording is already running, you **cannot** start a new one.

Stop trace recording

To stop a trace that is running, choose *Performance Trace* → *Stop Trace Recording* from the menu. If you do not stop a trace in this way, the system stops recording automatically at the time you defined.

You cannot select this menu option if no trace recording is running.

Save trace file

To save a trace file locally, choose *Performance Trace* → *Save Trace File* from the menu. When the trace file has been saved successfully, the SAP system automatically deletes the trace file from the system.

Display trace information

To display important key figures for the trace, choose *Performance Trace* → *Display Trace Information* from the menu. The system displays the following key figures for the trace:

- Start time
- Stop time
- Remaining time
- User
- File size in kilobytes

Delete trace file

When you save a trace file, it is automatically deleted from the SAP system.

However, if you want to delete a file without saving it, choose *Performance Trace* → *Delete Trace File* from the menu. The SAP system deletes the trace file from the system.

This is useful if the trace file has become too large.

4.4.7 Compressing InfoCubes

Use

When you load data into the InfoCube, entire requests can be added at the same time. Each of these requests has its own request ID, which is included in the fact table in the package dimension. This makes it possible to pay particular attention to individual requests. One advantage of the request ID concept is that you can subsequently delete complete requests from the InfoCube.

However, the request ID concept can also cause the same data record (where all characteristics are the same except the request ID) to appear more than once in the fact table. This unnecessarily increases the volume of data and affects system performance when you analyze data, since each time you execute a query, the system has to perform aggregation using the request ID.

You can eliminate these disadvantages by compressing data and bringing data from different requests together into one single request (request ID 0).

This function is critical, since you cannot delete compressed data from the InfoCube using its request ID. You must be absolutely certain that the data loaded into the InfoCube is correct.

Features

You can choose request IDs and release them to be compressed. You can schedule the function immediately or in the background. You can schedule compression as part of a process chain.

Compressing one request takes approximately 2.5 ms per data record.

With non-cumulative InfoCubes, compression has an additional effect on query performance. With non-cumulative InfoCubes, the marker for non-cumulatives is also updated. This means that less data has to be read for a non-cumulative query which reduces the response time. See also Modeling Non-Cumulatives with Non-Cumulative Key Figures [Page 420].

If you perform compression for a non-cumulative InfoCube, the compression time (including the time to update the markers) is about 5 ms per data record.

If you are using an Oracle database as your database, you can also execute queries on the relevant InfoCube while compression is running. With other manufacturers' databases, you get a warning message if you try to execute a query on an InfoCube while compression is running. You can execute the query once compression is finished.

If you do not want the InfoCube to contain entries with zero values for key figures (in reverse posting for example), you can run zero-elimination at the same time as compression. In this case, the entries where all key figures are equal to 0 are deleted from the fact table.

Zero-elimination is only allowed for InfoCubes that contain key figures with aggregation behavior 'SUM' exclusively. You cannot run zero-elimination for InfoCubes that contain non-cumulative values.

For non-cumulative InfoCubes, you can ensure that the non-cumulative marker is not updated by setting the indicator *No Marker Updating*. If you are loading historic changes to non-cumulative values into an InfoCube after initialization has already taken place using the current non-cumulative, you have to use this option. If you do not use this option, the results produced in the query are incorrect. For performance reasons, you should compress subsequent delta requests.

Activities

For performance reasons, and to save space on the memory, compress a request as soon as you have established that it is correct and is not to be removed from the InfoCube.

4.4.8 InfoCube Performance

Use

Here you can find the options that allow you to affect not only the load performance, but also the query performance.

Features

Database Indexes

With an increasing number of data records in the InfoCube, not only the load, but also the query performance can be reduced. This is attributed to the increasing demands on the system for maintaining indexes. The indexes that are created in the fact table for each dimension allow you to easily find and select the data. When initially loading data into the InfoCube, you should not create the indexes at the same time as constructing the InfoCube, rather only afterwards.

The indexes displayed are the secondary indexes of the F and E fact tables for the InfoCube. The primary indexes and those defined by the user are not displayed. The aggregates area deals with the associated indexes of the fact table for all aggregates of an InfoCube.

Checking Indexes

Using the *Check Indexes* button, you can check whether indexes already exist and whether these existing indexes are of the correct type (bitmap indexes).

Yellow status display: There are indexes of the wrong type

Red status display: No indexes exist, or one or more indexes are faulty

You can also list missing indexes using transaction DB02, pushbutton *Missing Indexes*. If a lot of indexes are missing, it can be useful to run the ABAP reports SAP_UPDATE_DBDIFF and SAP_INFOCUBE_INDEXES_REPAIR.

Deleting Indexes

For delta uploads with a large quantity of data (more than a million records), you should not align the database indexes of the InfoCube with every roll up, rather delete the indexes first, and then completely reconstruct them after rolling up.

Repairing Indexes

Using this function you can create missing indexes or regenerate deleted indexes. Faulty indexes are corrected.

Build Index

You can delete indexes before each data load and then rebuild them again afterwards. You can also set this automatic index build for delta uploads.

Aggregate Table Indexes

Just as for the indexes of the fact table, you can maintain the indexes of the aggregation tables. Maintaining the indexes of the aggregation tables affects all the aggregation tables of the InfoCube.

If the InfoCube contains more than 50 million records, you should, as a rule, refrain from deleting and recreating indexes.

Database Statistics

The database statistics are used by the system to optimize the query performance. You should keep the database statistics up-to-date for this reason.

It is recommended that you always update the statistics if you have loaded more than a million new records into the InfoCube since the last update.

You can automatically recalculate the database statistics after each load or after each delta upload.

Using the pushbutton *Check Statistics,* you can check the InfoCube to see if statistics exist. If no statistics exist yet, the status display changes to red. Using *Recalculate Statistics,* you can add the missing InfoCube statistics in the background.

You can determine the percentage of InfoCube data that is used to create the statistics. The percentage is set to 10% by default. The larger the InfoCube, the smaller you should choose

the percentage, since the demand on the system for creating the statistics increases with the change in size.

For up to 10 million entries in the InfoCube, you should set the percentage of InfoCube data, that is used for creating the statistics, to 100%.

Automatic deletion and building of indexes, as well as automatic calculation of database statistics and cause conflicts with automatic compression and rollup (see Automatic Subsequent Processing [External]). These processes lock one another. See Functional Constraints of Processes [External].

If you use process chains to guarantee serial processing of the process, these settings are not supported for the object. Then you have to include the automatic processes as process types in the process chain.

See also the example for Including InfoCubes in Process Chains [External].

4.4.9 Information Lifecycle Management

Use

In an operative BI system, the volume of data increases constantly. Constant changes to business and legal requirements mean that this data must be available for longer. Since keeping a large volume of data in the system affects performance and increases administration efforts, we recommend that you implement a data aging strategy.

Advantages of a data aging strategy:

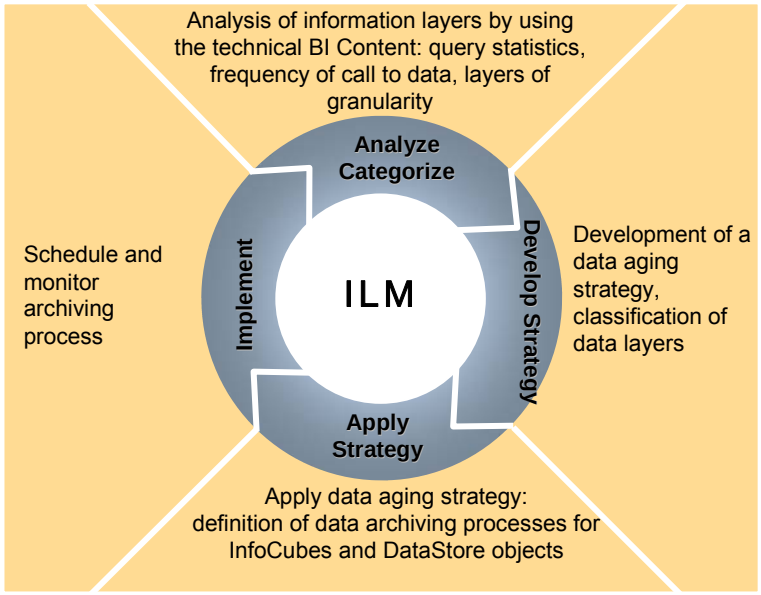
- Reduced costs for managing and saving data
- Improved performance of the BI system
- Efficient storage of historic data for comprehensive enterprise data warehousing

Prerequisites

You have loaded data to your InfoProvider, possibly over a long period of time.

Features

Information Lifecycle Management allows you to classify data according to how current it is and archive it or store it in near-line storage.



4.4.9.1 Data Archiving Process

Use

You can use ADK-based archiving or near-line storage for InfoCubes and DataStore objects. The central object is the data archiving process. When defining the data archiving process, you can choose between classic ADK archiving, near-line storage, or a mixture of both solutions.

The option you choose depends on your requirements and use of the data:

- You can archive the data based on ADK: this type of archiving is recommended for data that is no longer relevant to current analysis processes, but must remain archived for the legally-required storage period and must be provided for analysis in an emergency.
- You can store the data in near-line storage: near-line storage is recommended for data that you may still need. Storing historical data in near-line storage reduces the data volume of InfoProviders; however, the data is still available for BEx queries. You can also access the data archived in near-line storage from the query monitor. You do not have to load the data back into the BI system. The database partitions and the near-line storage partitions for an InfoProvider consistently reflect the total dataset.

  This technology is particularly useful for the Enterprise Data Warehousing layer, since the granular storage of data here can quickly result in large data volumes.
- You can combine ADK-based archiving with the storage of data in near-line storage: this type of archiving is recommended if you want an extra backup of your data in addition to near-line storage. Some third-party tools for near-line storage must also be connected to the ADK to allow the data to be stored. In this case, you must select both near-line storage and ADK-based archiving when you create a data archiving process.

Integration

Using a near-line storage interface, you can enhance your BI system with near-line storage solutions from third-party providers. In this way, selected SAP partners can integrate their near-line storage solutions into the BI system on the basis of ABAP APIs.

ADK-based archiving uses the Archive Development Kit (ADK). The ADK is a tool from SAP NetWeaver for developing archiving solutions that provides the runtime environment for archiving. Its main function is to read and write data to and from archive files.

More information: Introduction to Data Archiving (CA-ARC) [External]

Features

Using the data archiving process, you can archive and store transaction data from InfoCubes and DataStore objects. This function is not available for write-optimized DataStore objects.

The data archiving process consists of three main steps:

1. Creating the archive file/near-line object
2. Storing the archive file in an archiving object (ADK-based) or near-line storage
3. Deleting the archived data from the database

A data archiving process is always assigned to one specific InfoProvider and has the same name as this InfoProvider. It can be created retrospectively for an existing InfoProvider that is already filled with data.

In ADK archiving, an archiving object is created for each InfoProvider.

As with the role of the archiving object during ADK archiving, the near-line object addresses the connected near-line storage solution during near-line storage. It is also generated from the data archiving process for an InfoProvider. Near-line objects consist of various near-line segments that reflect different views of the respective InfoProviders and can also reflect the different versions of an InfoProvider.

4.4.9.1.1 Archiving Data

Use

Besides archiving data from InfoCubes and DataStore objects, you can also store it in near-line storage. Using the data archiving process you define whether the data is to be archived based on ADK, stored in near-line storage, or whether you want to use a mix of both. This data archiving process can be scheduled using either a process chain or the ADK. It can also be executed manually.

Prerequisites

You have already loaded data into the InfoProvider.

Procedure

1. You create a data archiving process for the InfoProvider from which you want to archive data.

   For more information, see Creating Data Archiving Processes [Page 741].

2. You generate and then schedule the archiving request. For more information, see:
   - For near-line and ADK-based data archiving processes: Scheduling Data Archiving Processes Using Process Chains [Page 743]
   - For purely ADK-based data archiving processes: Scheduling Data Archiving Processes Using ADK Archive Administration [Page 744]
   - For manual request processing: Creating and Executing Archiving Requests [Page 746]
3. You can load archived data back into InfoProviders.

   For more information, see Loading Archived Data Using a Data Transfer Process [Page 749]
4. In a query, you can access data stored in near-line storage.

   For more information, see: Displaying Stored Data in Queries [Page 749]

4.4.9.1.2 Creating Data Archiving Processes

Prerequisites

If you want to use third-party near-line storage, the connection for the third-party solution must be implemented and a near-line connection must be set up.

Procedure

1. Call the data archiving process maintenance (transaction RSDAP).
2. Select an InfoProvider for which the data archiving process is to be created.
3. Choose *Edit*. The data archiving process contains the same technical name as the InfoProvider. Enter a description.

Tab Page: General Settings

4. You can select whether you want to use ADK-based archiving, near-line storage, or both.

Tab Page: Selection Profile

5. You specify which data from the InfoProvider is selected for the archiving run.

   With reference to the selected time characteristic, the time slice archiving always creates time intervals that directly follow on from one another. These can be connected, with conditions, to additional, time-independent partitioning characteristics. You can use these characteristics to restrict the selection further.

   To allow for optimum prerequisites for the later use of queries on near-line storage, the selected time characteristic must also be highly relevant for the queries.

   For a DataStore object, you can only select a key characteristic as the partitioning characteristic in time slice archiving. The characteristic for the time slice generation can also be a non key characteristic. However, you must then select a separate partitioning characteristic from the key with a largely monotonous time reference and to which the time conditions can be applied (such as *Clearing Date* in the data part and *Document Number* in the key). At runtime, the system attempts to apply the time restriction to a

partitioning characteristic restriction. In the InfoCube, all characteristics always count as key characteristics.

Tab Page: Semantic Group

6. You can form semantic groups to archive the data in a sorted way.

   The system reads from the database after sorting according to grouping characteristics (the order is important). Records with the same specifications in the grouping characteristics are written to the archive as one data object.

   If no characteristic is selected, the storage in the archive is not sorted and technical criteria are used for classification (fixed size) into data objects.

Tab Page: ADK

7. Specify the logical file name.
8. Define a maximum size for the generated archive files. You must take the memory capacity of your storage medium into account. This can be restricted by the number of data objects. The limit that is reached first is the deciding one. During an archiving run, a new file is created when the maximum size is reached.
9. You can trigger delete jobs by events.
10. You can specify the storage system and delete sequence. You only need to specify the storage system and delete sequence if you are using a separate storage system.

Tab Page: Near-Line Storage

11. Specify a maximum size for the data package. You must take the memory capacity of your storage medium into account. This can be restricted by the number of data objects. The limit that is reached first is the deciding one. During an archiving run, a new data package is created when the maximum size is reached.
12. Activate the data archiving process. If the data archiving process is active, it is no longer possible to change many of the settings.

Result

Depending on how the archiving process is modeled, an archiving object or a connection to near-line storage is generated.

For ADK-based archiving, an archiving object is generated in the local system. The namespace for archiving objects consists of "BW" as the prefix, then a character for the object type ("C" = InfoCube, "O" = DataStore object A version), and finally the object name. Only seven characters are available for the name, since only ten characters are available for the whole name.

You can now schedule the data archiving process. More information:

Near-line and ADK-based data archiving processes: Scheduling Data Archiving Processes Using Process Chains [Page 743]

Purely ADK-based data archiving processes: Scheduling Data Archiving Processes Using ADK Archive Administration [Page 744]

Manual request processing: Creating and Executing Archiving Requests [Page 746]

4.4.9.1.3 Scheduling Data Archiving Processes Using Process Chains

Prerequisites

You have created a data archiving process.

Procedure

1. You are in the *Administration* functional area of the Data Warehousing Workbench. Choose **Process Chains**. The *Process Chain Maintenance Planning View* screen appears.
2. In the left-hand screen area of the required display component, navigate to the process chain in which you want to insert your data archiving process. Double-click to select it. The system displays the process chain plan view in the right-hand screen area.

   If no suitable process chain is available, create a new process chain. For more information, see Creating Process Chains [Page 402].
3. In the left-hand screen area, choose **Process Types.** The system displays the available process categories.
4. In the **Data Target Administration** process category, choose the application process type **Archive Data from an InfoProvider**.
5. Insert the application process type into the process chain using drag and drop. The dialog box for inserting a process variant appears.
6. Create a process variant and confirm your entries. An additional dialog box appears.
7. Select your data archiving process.
8. Make the required settings:
   - Under **Process Flow Control**, you can either continue open archiving processes or specify whether the archiving process is to run until a specific status only. For more information, see Status of Archiving Request [Page 747].
   - Under **Relative Time Restrictions**, you can specify the area to be archived. You can schedule periodic archiving runs without having to adapt a variant with each run using a relative time condition in the form of a minimum age.

     The relative time restriction is changed into a selection condition for the time slice characteristic, depending on the start time of the write process. An intersection is created using the absolute time conditions for the time slice characteristic and the selection conditions for the other characteristics. This intersection is then used to derive the already archived area and the selection conditions from other archiving requests that are still open. The resulting selection condition is stored with the newly created archiving request and is used for both the selection of data in the write phase and the delete phase.
   - Under **Further Restrictions**, you can select values of your partitioning characteristics.
9. Confirm your entries. At runtime for the process step, an archiving request is generated from this variant, which controls the entire archiving run.

10. Save and activate the process chain. The process chain starts according to the selections in the start process.

The InfoProvider to be archived is not locked for data loading during the entire process. The lock is only necessary for a short period before the data is copied. After this, only the previously archived data areas and the area that is being archived are locked.

4.4.9.1.4 Scheduling Data Archiving Processes Using ADK Archive Administration

Prerequisites

You have created a data archiving process with ADK-based archiving.

Procedure

Archiving Data

1. Call archive administration (transaction SARA).
2. Create an archiving variant. See also Maintenance of Variants [Page 745].
3. Schedule the archive run; this allows you to test the run in test mode first, before starting the actual archive run in productive mode.

You can find a detailed description of the procedure under Scheduling Preprocessing Programs [External] and Creating Archive Files [External].

The InfoProvider to be archived is not locked for data loading during the entire process. The lock is only necessary for a short period before the data is copied. After this, only the previously archived data areas and the area that is being archived are locked.

Deleting Archived Data

When the archiving run has successfully ended, you can delete data from the database using the archiving run selection criteria.

You are in the initial screen of Archive Administration. Choose *Delete*.

The InfoProvider is locked for other actions until the archived data has been deleted from the InfoProvider.

For a more detailed description of the procedure, see Deleting Archived Data from the Database [External].

For more information about archive administration, see Archive Administration [External].

4.4.9.1.4.1 Maintenance of Variants

Use

When you schedule jobs for executing archiving programs, such as the preprocessing, write, or delete programs, you must specify a variant. The purpose of this variant is to specify the scope of data to be processed and to allow the user to make the necessary settings for data processing.

Features

Relative Time Restrictions

Under *Relative Time Restrictions*, you can specify selection conditions that are relative to the date for the execution of the write program.

Only Records Older Than: You can determine the upper limit of the time slice. To do this, you specify a time period relative to the current date. The time unit for the time period can be specified in days, weeks, months, quarters, years, posting periods, or fiscal years.

Only Complete: The upper limit can also be rounded down to the end of a time period by selecting another, additional time unit, so that, for example, only complete months or years can be archived when a date field is selected as a selection characteristic for the time slice. The time slice defined in this way contains the complete time period before the upper limit calculated.

If you have specified a relative time restriction and confirmed it with *Enter,* the selection condition is displayed in the lower row.

Further Restrictions

Under *Further Restrictions*, you can specify selection conditions for additional characteristics that you chose under *Selection Profile* in the data archiving process. For the actual selection of the archiving run, the intersection of the relative time restrictions and absolute time restrictions is taken.

Example

You made the following entries in the variant maintenance (February 2006):

Only Records Older Than	2 Years
Only Complete	Quarters
Calendar Day	≤ 31.12.2003 (this is the automatic display for the selection option)

With this selection, only data from complete quarters and that is older than two years is archived. At the current point in time, everything older than or equal to 31.12.2003 is archived.

4.4.9.1.5 Administration of Data Archiving Processes

Use

When administrating a data archiving process you can manually (without using process chains) create, execute, and reload the archiving requests, as well as make them invalid.

Prerequisites

You have created and activated a data archiving process for an InfoProvider.

Features

In administration, you can create an archiving request for active data archiving processes. With data archiving processes that store the data in near-line storage, you can create a request, lock the selection area, write the data to near-line storage, verify the storage, and delete the data from the InfoProvider. If, for any reason, your requirements have changed, at any point before the verification, you can mark the request as invalid. Once the request has completed all the individual steps involved and the overall status is on green, you can reload the data back into the InfoProvider if required. This is only necessary, however, in exceptional cases.

In ADK-based data archiving processes you can only execute the archiving process in one step.

See also:

Creating and Executing Archiving Requests [Page 746]

4.4.9.1.5.1 Creating and Executing Archiving Requests

Use

You can create an archiving request manually without using a process chain.

Prerequisites

You have created and activated a data archiving process for an InfoProvider.

Procedure

1. You are in the *Modeling* functional area of the Data Warehousing Workbench. Choose *Manage* in the InfoProvider tree in the context menu for your InfoProvider.
2. Choose the *Archiving* tab page. This is only displayed if there is an A version of the data archiving process.
3. Choose *Create Archiving Request.*
4. Make the required settings:
   - Under *Relative Time Restrictions*, you can specify the area to be archived.
   - Under *Process Flow Control*, you can either continue with the open archiving processes or specify whether the archiving process is to run until a specific status only. For more information, see Status of Archiving Request [Page 747].

For data archiving processes without an ADK connection, you can also choose between the *Test Mode* and *Productive Mode*. Only tests are run in the test mode. No data is actually archived.

- Under *Further Restrictions*, you can select values of your partitioning characteristics.

5. Choose *Execute*.
6. Choose *Back*. The generated archiving request is displayed.
7. You start other processing steps using the status columns. The status of the request depends on what you specified under *Process Flow Control.* You can execute the rest of the steps by choosing the pushbutton for the last status entry. For more information on the different statuses, see Status of Archiving Request [Page 747].

Result

If the archiving request has run in the productive mode and the status is green, the data has been archived. If you then discover that you still need the data in the InfoProvider, you can reload it. This is only to be done, however, under exceptional circumstances.

Reload Request

Using the pushbutton in the *Overall Status* column for the completed request you can generate a reload request, where only near-line storage is involved. If the archiving took place with the ADK, the reload process must be triggered using archive administration.

The reload request is directly related to the archiving request for which it was created. If the process is terminated, you can mark the reload request as invalid and create another reload request.

4.4.9.1.5.2 Status of Archiving Request

In the administration of an archiving request, you can execute additional processing steps using the respective last status. The information below explains what is executed when a particular status is set and what you need to consider when doing this

1. Lock Status

You can lock the selected data area of the archiving request to prevent any changes. This is necessary before data archiving begins. This step can be executed in dialog mode.

2. Copy Status

The data to be archived is copied into near-line storage or the archive. This step is to be executed in the background.

When the write phrase is complete, the archiving request can be marked as invalid if your requirements have changed in the meantime. The data to be archived is still available in the InfoProvider.

3. Verification Status

In the verification phase, the system checks that the write phase was successful and that the data can be deleted from the InfoProvider.

The archive file assigned to the process is opened in test mode and read completely. This ensures that the archive file is complete and can be accessed. If the check is successful, this is noted in a separate status table for the archive file.

4. Deletion Status

When the archived data is deleted from the InfoProvider, the archiving process is complete.

The archived data is deleted from the InfoProvider with the same selection conditions used to copy the data from the InfoProvider. Once the data has been successfully deleted from the InfoProvider, all the archive files from the archiving run are confirmed using the ADK. The archive files are opened in productive mode and closed again immediately. When storing the data in near-line storage, the process is similar.

5. Overall Status

When all the steps of the archiving request have been completed, you can no longer change the status of the request. You can only reload the data. To do this, you can create another request for the reload.

4.4.9.1.5.3 Archivierungsrequest auf ungültig setzen

Verwendung

Aus verschiedenen Gründen kann ein Archivierungsrequest während der Schreib- oder Verifikationsphase den Status *Gelb* erhalten. Sie können in diesem Fall den Request automatisch oder manuell auf ungültig setzen, um einen neuen Archivierungsrequest anzulegen und die Archivierung dieser Daten neu zu starten.

Voraussetzungen

Funktionsumfang

Automatische Request-Invalidierung

Sie können in der Administration eines InfoProviders bzw. in der Archivadministration (Transaktion SARA) zu einem Archivierungsrequest die automatische Request-Invalidierung einschalten.

Mit der automatischen Request-Invalidierung wird ein Archivierungsrequest, der während der Schreib- oder Verifikationsphase den Status *Gelb* erhält, automatisch auf ungültig gesetzt. Damit können Sie verhindern, dass ein InfoProvider für weitere Aktionen gesperrt ist. Insbesondere bei einem DataStore-Objekt mit einem Zeitscheibenmerkmal aus dem Datenteil, das trotzdem als primäres Partitionierungsmerkmal im Archiv verwendet wird, bleibt sonst das DatenStore-Objekt für die Aktivierung neuer Requests und für andere Aktionen, die zur Veränderung der Daten in der aktiven Tabelle führen, gesperrt.

Manuelle Request-Invalidierung

Sie können einen Archivierungsrequest in der Administration eines InfoProviders manuell auf ungültig zu setzen. Sie können damit sowohl einen Archivierungsrequest, der während der

Schreib- oder Verifikationsphase den Status *Gelb* erhält, automatisch auf ungültig setzen, als auch einen Rückladerequest, der abgebrochen ist.

4.4.9.1.6 Loading Archived Data Using a Data Transfer Process

Use

Data from an InfoProvider that is archived on an ADK basis using a data archiving process or that was stored in near-line storage can be loaded into other InfoProviders. Using a data transfer process with the archived InfoProvider as the source you can get a connected InfoProvider directly from the archive.

Procedure

1. You create a data transfer process for an InfoProvider that is connected to the archived InfoProvider. The data transfer process uses the archived InfoProvider as the source and the connected InfoProvider as the target.

   See: Creating Data Transfer Processes [Page 322]

2. In the data transfer process, you can decide whether you want to extract from the archive by choosing the appropriate option on the *Extraction* tab page under *Extraction from*. You can extract only from either the archive or the dataset contained in the InfoProvider. You do not need to select the relevant archiving runs.

   The filter conditions of the data transfer process are mapped to the selection conditions of the existing archiving runs before the archive is accessed, meaning an intersection is calculated. If they overlap, the relevant archiving request is opened automatically for extraction.

3. Execute the data transfer process.

4.4.9.1.7 Displaying Stored Data in Queries

Use

In a query, you can access data stored in near-line storage.

Prerequisites

In the Query Designer, you have defined a query based on an InfoProvider for which data is already stored in near-line storage.

Procedure

4. Call the query monitor by choosing *Business Explorer* → *BEx Monitor* → *Query Monitor* in the SAP Easy Access Menu.
5. Select your *Query*.
6. Choose *Properties*.
7. Select the *Read Near-Line Storage As Well* indicator.
8. Choose *Continue*. You changes are saved.

Result

When a query is executed, the data in near-line storage is also read and displayed.

4.4.9.2 Id Data Archiving Concept

Use

We recommend that you use the new data archiving concept with new objects. This allows you to use ADK-based archiving or near-line storage. For more information, see Data Archiving Process [Page 739].

The old data archiving concept is still available, but will not be developed further.

Data archiving enables you to simplify InfoCube and DataStore object administration and improve system performance.

The Archive Development Kit (ADK) is used for this purpose. ADK is a tool from SAP NetWeaver for the development of archiving solutions that provides the runtime environment for archiving. Its main function is to read and write data to and from archive files. The ADK guarantees that the archived data is both release- and platform- independent.

See also Introduction to Data Archiving (CA-ARC) [External].

Data archiving is used in BI for data that is no longer needed in running analysis processes, but is still important in the creation of new InfoProviders, or which may need to be analyzed once more in exceptional circumstances.

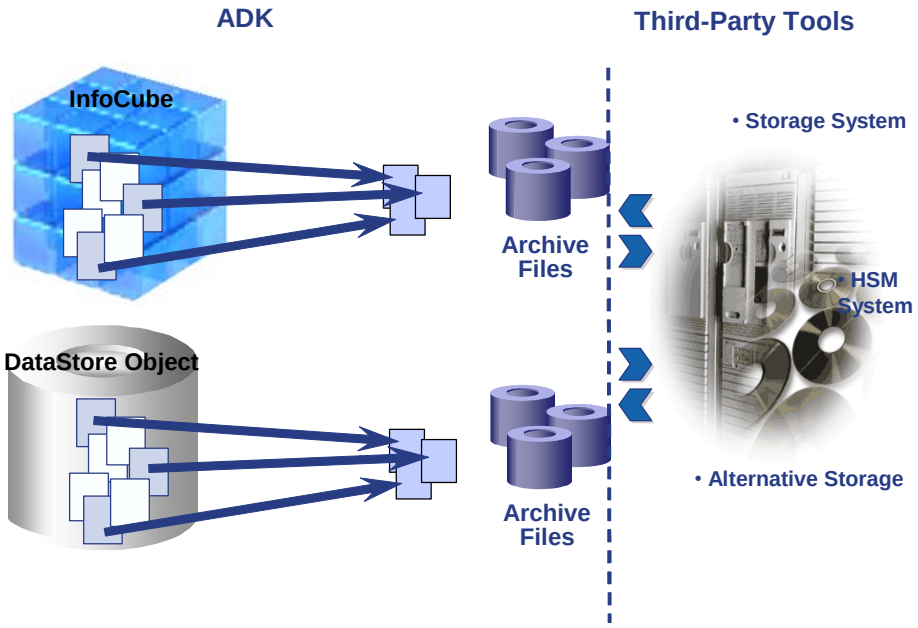
Features

Data archiving allows you to archive data from InfoCubes and DataStore objects.

You can access the function *Archiving* via the main menu in InfoCube and DataStore object maintenance. Here you can maintain the properties of an archiving object. You can also select both the selection characteristics and the maximum size for an archive file. An archiving object is generated from these properties when a data target is archived. You access the administration for the respective archiving object using the context menu for the individual InfoProvider. You can schedule archiving sessions and call up further archiving functions from here.

The archive object Administration entry only appears in the context menu when an archive object has previously been generated for the InfoProvider.

An archiving session consists of a write-, delete-, and a storage phase. Archive files are stored during the write phase for a particular archive object, deleted from the database after the verification phase, and then stored in a storage system according to its definition in Customizing.



Special archiving run features:

- The selection criteria control the archiving. No additional IMG entries (for example, document age) are required, as in SAP NetWeaver data archiving.
- Data is not stored in multi-dimensional tables for InfoCubes, but in a flat structure.
- Only the contents of the table for the active data are stored for DataStore objects and not change log content.
- Archiving information is not transferred to the connected InfoProviders with data that has already been updated. That is, data from nearby InfoProviders remains where it is and is not archived.
- The delete phase is based in the selected archiving session selection criteria. Here, existing aggregates are either adjusted or newly created. The delete phase can take place at a different time to the write phase.

Archived data can also be extracted again from the BI archives. There is also the option in the Scheduler to read data from archives and call up input help for existing archive files.

4.4.9.2.1 Data Archiving Process Flow

Purpose

You can use data archiving for data in InfoCubes and DataStore objects.

Process Flow

For general information about the process, see Schematic Data Archiving Process [External]. For the specific process flow in a BI system, refer to the following process description.

1. To archive data from an InfoProvider, you first need to create an archiving object for the InfoProvider.

   See also: Archiving Object [Page 755]

2. Archiving is carried out in the Archive Administration.

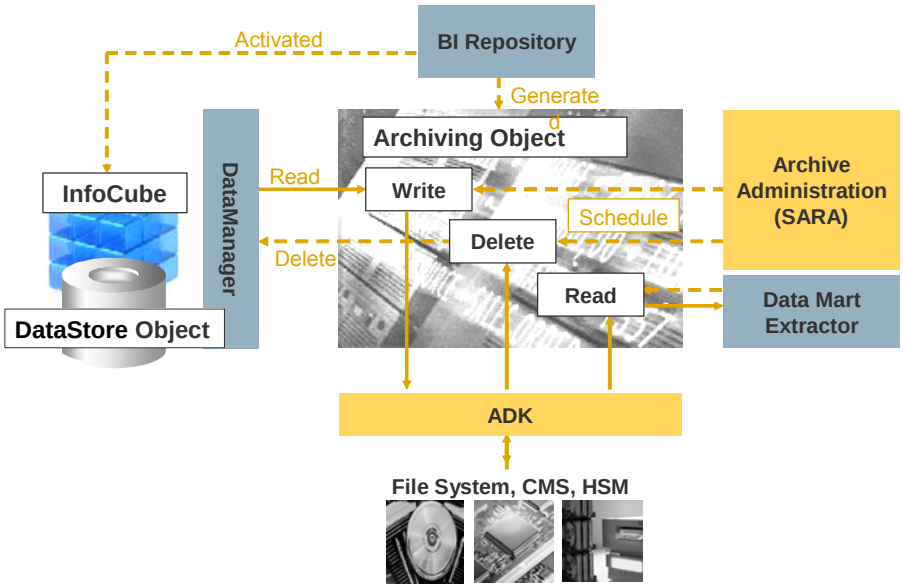
   You can call up the following commands from here.

   - Maintain archiving variants.
   - Schedule archiving run; here you can firstly start a test run before starting the actual archiving run.
   - Schedule a delete run
     - Store and retrieve from a storage system.

   Also refer to the detailed documentation in Archive Administration [External].

   Special technical features in the BI system:

   The following display shows the BI data archiving architecture:



Description of special technical features in the BI system:

Variants maintenance:

To schedule the execution of the write program, you first need to create a variant. The structure of the selection screen depends on the archiving method you chose when defining the archiving object. With the archiving of time slots (time periods with beginning and end points), you can only use selection conditions for the selected time characteristic. See also: Time Restrictions [Page 759] In doing so, maintenance is kept to a minimum.

Writing to the archive

Read or delete access to the data from the archiving processes is made possible using the data manager interface. When data is read from InfoCubes, it is transformed from the star-shaped table structure into a flat format that only contains the actual

characteristic attributes. Therefore, the archive is not affected by any reorganization of the IDs in the star schema. The disadvantage is an increase in the volume of data that is at least partially cleared again by means of the data compression in the archive file.

For each archiving object, an executable ABAP program is generated to write to the archive. The program generates and processes archiving requests.

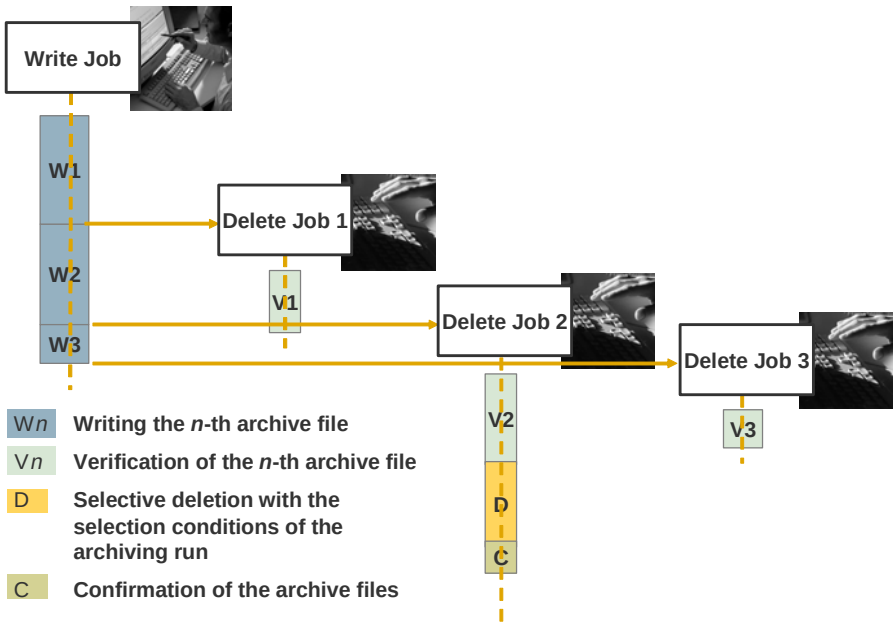
First of all, it generates an archiving request. The selection criteria and archiving run information is stored in the archiving request, and a status is administered. The system uses the InfoProvider interface to read InfoCube- and DataStore object data according to the grouping characteristics that were set. The data is written to archive files in the I structure (no SIDs, no navigation attributes, table-enabled field names) for InfoCubes, and in the structure of the A table (active data table) for DataStore objects.

Deleting archived data:

A separate background process is started for each archive file to delete the archived data. Depending on the archiving object settings, it is either automatically started in the write phase, started manually in Archive Administration or is triggered by a particular event. Each productive delete process has three steps:

1. In the first step, the archive file is verified in the test mode. The system checks here whether the archive file is complete and whether it can be accessed. Successful verification of the file is stored persistently in the corresponding archive request. The background process is ended if the write phase is still running, or if the system has not verified all archive files in the associated archiving run.
2. The second step begins when the archiving run write phase has ended successfully, and all generated archive files have been successfully verified. In this step, data is deleted from the database with the selection criteria of the archiving run. An optimal delete strategy is displayed for the selection criteria (also see Selective Deletion from an InfoCube [External] and from an DataStore Object [External]). For InfoCubes, this step also involves the aggregates being adjusted.
3. When the data has been successfully deleted, all archive files in the archiving run are confirmed. This is the third step. As a result, the data is seen as successfully deleted with regards to the ADK.

   Here is a schematic display of the deletion process:



For more information on the strategies that the system chooses for deletion, read the Background Information [Page 754].

3. You can use the Extractor Checker (transaction RSA3) to check an archive file.

   Also refer to Check Archive File [Page 766].

4. You can extract data from BI archives in order to reload it into the InfoProvider.

   We do not recommend you reload data directly into the original object. As an alternative, you can use the archive connection for the Export DataSource of an InfoCube or DataStore object. Archived data can then be extracted from the archive of the original object and loaded into an InfoProvider with the same (or a similar) structure using the data mart interface. Reporting permits a combination of data that has yet to be archived with data that has been extracted from the archive using the MultiProvider function.

   Also refer to Reload Data from Archive File [Page 766].

You can find detailed step-by-step instructions under Archiving Data [Page 760].

For more information about possible errors during the archive run and their removal, read the Background Information [Page 755].

4.4.9.2.1.1 Deletion Strategies

Depending on the selection conditions, different deletion strategies are used for the data volume to be deleted as well as the properties from the database tables.

- If the complete InfoProvider is deleted based on the selection conditions from the archiving run, then the affected tables in the database are deleted with a DDL command and are recreated.
- If the section of records to be deleted has exceeded a specific threshold value (for example, ten percent), then a copy of the table is created with the selection conditions

that complement the archiving run. The table is deleted afterwards, and the new table is renamed to the name of the old table.

- If the section of records to be deleted remains below the threshold value, it is selectively deleted in the database table with DML commands (Data Manipulation Language).
- If the fact table for an InfoCube is partitioned, the deletion occurs under specific conditions by deleting the complete partitions (drop partions).

The same strategies apply for InfoCube aggregates as they do for the fact tables with one difference: If you delete aggregates, only the deletion conditions for the characteristics that are also used in the aggregate are shifted. As a result, too much is deleted from the aggregate in some cases. This is corrected when the aggregate is completed by means of aggregating the difference quantity from a more detailed aggregate or, if no such aggregate exists, from the fact table.

4.4.9.2.1.2 Typical Errors and Their Removal

Depending on the phase in which an error occurs in the archiving process, different reactions are possible:

- If the write program terminates after the first archive file was already opened, then you have to explain first why this archiving run is invalid before you can restart the archiving process. By setting this property, you also removed the set locks for the InfoProvider when starting the write program.
- If the writing phase is successfully completed, and the error occurs in the verification phase, then you can do two things depending on the cause of the error, such as a temporary problem when accessing the archive file. You can either restart the delete program, or you can declare the archiving run invalid and can reset the archiving process with the same selection criteria.
- If an error occurs in the actual deletion phase or in the subsequent confirmation phase, you have to remove the cause of the error and, afterwards, reschedule the deletion program.

4.4.9.2.2 Archiving Object

Definition

A logical object containing related business data in a database that can be read from the database with a write program, and, after successfully being archived, can be deleted from the database using a corresponding delete program.

Use

To archive data from InfoProviders, you first need to create an archiving object for the InfoProvider. InfoCubes and DataStore objects usually contain a self-enclosed dataset for a specific business-related area. For this reason, an individual archiving object is generated for each InfoProvider. Only one archiving object can be created for each InfoProvider.

First of all, maintain the metadata of the archiving object in the InfoProvider The actual object is not generated until the InfoProvider is activated.

You can also create the archiving object afterwards for an InfoProvider that already exists and is filled with data.

Structure

The following steps are required to define an archiving object:

Select Data

Time slot

You can enter a time here when the system is to carry out archiving. If you choose *None,* you need to choose selection characteristics in the lower half of the screen instead. By choosing *By Characteristic,* you can use the dropdown list box to select just a time characteristic or a characteristic that references a time characteristic.

Choose this time slot mode if you want to archive regularly. In this way, the selection screen of the write program changes also, whereby you reduce the administration workload, because a new variant does not need to be created for each archiving run. In the maintenance for variants, you can specify whether you want a relative and/or absolute time restriction. See also: Time Restrictions [Page 759]

Only for DataStore objects: If the *Protect Archived Area* indicator is set, the system checks new data before it is activated to see if these data records that are in a time slot have already been archived. If this is the case, the system terminates activation and writes the affected archiving runs to the archiving log. For additional procedures you have two alternatives:

- To activate the data successfully, you can reload the archiving runs, which were specified in the log, into the table of active data for the DataStore object by using the archiving administration (see Reloading Archives [External]). Activation can now be started again.
- Or delete the affected requests from the activation queue for the DataStore object, modify the update rules so that the affected data records are skipped, and load and reactivate the requests.

The protection of archived data areas is also possible with a time characteristic, which is not a key characteristic of the DataStore object. In some cases, however, you might need to check whether the protection is worthwhile or not. For example, protection cannot be ensured if the value of the time characteristic can be subsequently enlarged in a data record (and, in doing so, lies outside of the protected area). A time characteristic in the data portion that is derived from the archiving status in the source system can, however, be by all means useful.

This key figure needs to be set for archive administration to be able to allow data to be reloaded directly into the active data table for a DataStore object.

Choose selection characteristics

The characteristic you have specified for time selection is displayed here. You cannot choose any additional selection characteristics.

If you have not specified a characteristic for time selection, you must set selection characteristics here. One of these characteristics must be a key characteristic and must

display a measurement for the age of the data. The following criteria are available for the selection:

- If the InfoProvider is partitioned, the selection characteristic must be identical to the partition characteristic.
- If the InfoProvider is not partitioned, you must select a time characteristic from the key.

If neither of these conditions is fulfilled (this is possible for DataStore objects, for example), additional characteristics of the InfoProvider can be used as selection characteristics for archiving sessions. Examples are the characteristics such as product (when a product is being phased out, for example) or company code/company (when an organization sells or unloads a legal unit).

The archiving method cannot be changed after the first activation run. For more information, see Changing or Deleting an Archiving Object [Page 758].

File Structure

Define a maximum size for the generated archive files. You must take the memory capacity of your storage medium into account. The maximum size is specified in Kbytes. It can be restricted by the number of data objects (see below). The limit to be reached first is the determining one. During an archiving run, a new file is created when the maximum size is reached.

Define your data object in the lower half of the screen. A data object is the smallest indexable and accessible unit in the archive. A data object for transaction data can be created using a quantity of records that have the same values (= grouping characteristics) with respect to the key characteristic selection. The system reads from the database after sorting according to grouping characteristics (the order is important). Records with the same specifications in the grouping characteristics are written to the archive as data objects. In some circumstances, a data object defined in this way can be subdivided again into several data objects to limit size. The grouping characteristics thereby form the basis of a possible archive indexing.

The data is then sorted according to these characteristics and the same records are grouped in a data object. If no characteristics are selected here, data is written to the archive file as unsorted.

Store

Here you must specify the logical file name, storage system and delete sequence. You only need to specify the storage system and delete sequence if you are using a separate storage system.

Generating an Archiving Object

An archiving object is generated locally from the archive-specific attributes when activation takes place.

The name range for archiving objects consists of: ‘BW’ as the prefix, then a character for the object type (‘C’ = InfoCube, ‘O’ = DataStore object A version), then comes the object name, for which only seven characters are available, since only ten characters are available for the whole name.

A mapping table is created. This assigns the BI InfoProvider an unambiguous name for an archiving object.

When the InfoProvider is deleted, the archiving object and the mapping table entry are also deleted. In addition to the archiving object, programs to write to the archive and to delete from Archive Administration are generated.

Integration

Deleting an InfoProvider also deletes the archiving object assigned to it. This also deletes links to the archive files. A confirmation prompt appears before you delete the InfoProvider if the archive contains data.

When deleting the content of an InfoProvider, only data from the database tables is deleted. Links to the archive files remain in place.

See also:

Change or Delete an Archiving Object [Page 758]

Archiving Object [External] (SAP NetWeaver)

4.4.9.2.2.1 Deleting or Changing an Archiving Object

Features

When changing or deleting an archiving object for an InfoCube or an DataStore object, there are the following restrictions:

1. If data has already been archived, certain BI-specific properties of the archiving object can no longer be changed:
   - The archiving method, that is, whether it is archiving with time slots or with free characteristic selection, can no longer be changed.
   - Characteristics that were selected as selection characteristics or were selected for the definition of the data object cannot no longer be undone.
   - The setting for area protection in a DataStore object can no longer be changed.
2. An archiving object generated once can only be deleted together with the InfoProvider.

Workaround for Subsequent Changing of Archiving Objects

For DataStore objects with an active protection of already-activated data areas, the changeability can be restored when you completely reload the already executed archiving run using the appropriate function in the archive administration.

Workaround for Deleting Archiving Objects

Deleting an archiving object independently of the associated InfoProvider is not supported.

In exceptional cases you can delete an archiving object independent of the assigned DataStore object or InfoCube with the program RSARCH_ARCH_OBJ_DELETE (execute using transaction SE38 or SA38). However, all administrative data about already-generated archive files is also lost. The program, however, should only be used when you are sure that you no longer need the already-archived data or that you have already completely reloaded the data into the InfoProvider or an alternative InfoProvider.

We strongly advise you not to delete an archiving object in production operation for data that has already been successfully archived.

Be especially careful when you execute the deletion program in a development system and then transport the associated InfoProvider into a production system. During import into the target system, there is no check, as to whether there are

already archiving runs for the archiving object, and the reference to the InfoProvider for an archiving object that is already present is lost (the archiving object and its archiving runs are not deleted, however).

You should only delete an archiving object when the InfoProvider

- was not yet transported further with the archiving object,
- was not yet archived in the supplied target systems or
- the archiving object was also deleted before in the target systems (under the restrictions listed above).

4.4.9.2.3 Time Restrictions

Use

Using relative and absolute time restrictions reduces the maintenance workload for data archiving. Variants with relative time restriction are designed for periodic scheduling of archiving runs (for example, through a periodic event).

Prerequisites

You have selected a time characteristic under a time slot when in the definition of the archiving object. See also: Archiving Object [Page 755]

Features

Under Relative Time Restrictions, you can specify a selection condition (for the selected time characteristic) that is relative for the data for the execution of the write program.

Only Records Older Than: Here you can determine the upper limit of the time slot. For this you specify a time period relative to the current date. The time unit for the time period can be specified in days, weeks, months, quarters, or years.

Only Complete: The upper limit can be rounded down here by selecting another, additional time unit for the end of a time period. In doing so, you can also archive, for example, only complete months or years when selecting a date field as a selection characteristic for the time slot. The time slot that is defined this way initially contains the complete time period before the upper limit that was calculated in this way. Optionally (when protecting the archived data areas in an ODS object, this option is fixed and set), time slots that were already archived can appear from the selection. In this case, the lower limit of the time slot normally comes from the lower limit of the time slot for the preceding archiving run.

Absolute Time Restrictions: Here you can specify additional selection conditions for the time characteristic. For the actual selection of the archiving run, the intersection of the relative and absolute time restrictions is then taken. In doing so, absolute time restrictions can be used especially with exclusive selection conditions to exclude specific time periods from the selection. If you initially want to leave the time period for relative time restrictions (= all values), then only the absolute time restrictions are valid.

If you have specified a relative time restriction and confirmed it with *Enter,* the condition is displayed in the bottom row.

For example: You have made the following specifications (in 2002):

Only records older than:	2 years
Only complete	Fiscal years
Fiscal year	< 2000 (this is the automatic selection option display)

This selection has the result that only data is archived from complete fiscal years that is older than two years. From the current time, this would be all records that were loaded prior to 2000.

With *Absolute Time Restrictions,* the time restriction you selected is displayed. You can specify an absolute time restriction. This is independent of the current time.

For example, you have made the following specifications:

Fiscal year	1998 to 2000

This selection results in data being archived that was loaded between 1998 and 2000, independently of the current date.

If you have specified both relative and absolute time restrictions, the intersection of both is calculated.

For example, you have made the following specifications:

Relative Time Restriction:

Only records older than:	2 years
Only complete	Fiscal years
Fiscal year	< 2000

Absolute Time Restriction:

Fiscal year	1998 to 2000

This selection means that only data from 1998 and 1999 is archived.

4.4.9.2.4 Archiving Data

Procedure

Creating the archiving object

1. You are in the maintenance for the InfoProvider from which you want to archive data. From the main menu, choose *Extras→ Archive...* To create an archiving object, confirm the subsequent dialog box.
2. On the *Select Data* tab page, either select a characteristic for time restriction under *Time Slots,* or, alternatively, set the selection characteristics. You can do this by dragging the required selection characteristics into the left-hand side of the screen with Drag&Drop , or you can use the arrow keys. See also: Time Restrictions [Page 759]

For data archiving from non-cumulative InfoCubes, you need to be aware of some special features. See also Archiving Data from Non-Cumulative InfoCubes [Page 762].

3. On the *File Structure* tab page, specify the maximum size of the generated archive files and define the data object. Do this by dragging the required characteristics with Drag&Drop into the left-hand side of the screen, or by using the arrow keys. The data is then sorted according to these characteristics and written to the archive file.
4. On the *Folder* tab page, specify the logical file name, storage system and delete sequence. You only need to specify the storage system and delete sequence if you are using a separate storage system.
5. On the *Delete* tab page, choose your delete options.
6. Choose *Continue*. You are now back in the maintenance screen for the InfoProvider.
7. Save and activate your InfoProvider. Your archiving object is now generated.

You can find more detailed information under Archiving Object [Page 755].

2. Executing Archiving

1. You are in the InfoProvider tree. From the context menu for your InfoProvider, choose *Archive Administration.* The Archive Administration Initial Screen now appears (transaction SARA).
2. Create an archiving variant.
3. Schedule the archive run; this allows you to test the run in the test mode first, before starting the actual archive run in the productive mode.

You can find a detailed description of the procedure under Scheduling an Initial Run [External] and Generating Archive Files [External].

Detailed technical information can be found under Data Archiving Flow [Page 751].

3. Deleting Archived Data

When the archiving run has successfully ended, you can delete data from the database using the archiving run selection criteria.

You are in the initial screen of Archive Administration. Choose the pushbutton *Delete*.

The InfoProvider is locked for other actions until the archived data has been deleted from the InfoProvider.

You can find a more detailed description of the procedure under Deleting Archived Data from the Database [External].

Detailed technical information can be found under Data Archiving Flow [Page 751].

You can find detailed information about archive administration in the documentation for Archive Administration [External].

4.4.9.2.4.1 Archiving Data from Non-Cumulative InfoCubes

Use

In order to archive data from InfoCubes with non-cumulative values, you need to be aware of a number of peculiarities in the procedure.

For more information on the structure of InfoCubes with non-cumulative key figures, see Non-Cumulatives [Page 420].

Features

There are two ways of archiving data from a non-cumulative InfoCube - time slot archiving and archiving with selection characteristics.

Time slot archiving (recommended)

As an archiving method, you must choose for a non-cumulative InfoCube, time slot archiving for the time reference characteristic. When you recreate an archiving object for a non-cumulative InfoCube, this characteristic is also suggested by the system.

Only archive non-cumulative changes up to a specific point in the past. This reduces the volume of data in the InfoCube. The marker (= end non-cumulative) remains unchanged.

In the variants maintenance screen, in the archive administration, under *Non-Cumulatives*, choose the *Movements Only* option.

You are only able to archive the marker if you also archive all of the non-cumulative changes. Subsequently, the contents of the non-cumulative InfoCube are archived completely. In the variants maintenance screen, in the archive administration, under *Non-Cumulatives*, choose the *Movements and End Non-cumulative* option.

You must archive the non-cumulative changes (for all times) first, and then archive the marker in a separate archiving run, so that the non-cumulative changes and the marker are stored separately in the archive.

The advantage of time slot archiving is that archiving runs can be carried out periodically using a single variant with a relative time restriction. The alternative archiving method described below does not have this advantage.

Archiving with selection characteristics

In some cases, archiving based only on time slots does not have the required effect of reducing the volume of data in the InfoCube. This is the case, for example, if the non-cumulative bearing objects are subject to high fluctuations, meaning that data is constantly being loaded for new objects and old objects are made obsolete. With time slot archiving, the end non-cumulative remains for all objects until all of the data in the InfoCube is finally archived, even if all the non-cumulative changes for an object have been archived already.

In this case, you want to archive and delete completely any objects that are no longer active in the InfoCube. This means archiving the marker as well. For other objects that are still active, you only need to archive the non-cumulative changes from the past, which means the marker is retained.

When you define the archiving properties for a non-cumulative InfoCube, do not therefore choose the time slot archiving method. Instead, select individually the characteristics that are relevant to restricting archiving runs. The time reference characteristic should also be included in this selection whenever possible.

Using this archiving method, for each archiving run, you need to create selection conditions in a variant. In the variant for the archiving run, in addition to the selection conditions, you can

also specify for the selected characteristics whether or not non-cumulative changes and marker data is archived. If you choose to archive marker data, be aware that the time selection contains the maximum value for time.

Reloading data from the archive

As a general rule, SAP recommends that you do not reload archived data back into the InfoCube that it originally came from. It is better to reload the archived data into a structurally identical copy of the original InfoCube instead. For the purposes of analysis, both InfoCubes can be combined using a MultiProvider. This applies in particular to non-cumulative InfoCubes, but there are a few implications here as well.

The procedure for reloading archived data is different for non-cumulative changes and for marker data. Note the following:

- Marker data and non-cumulative changes must always be reloaded from the archive in separate requests.
- Marker data must be reloaded before the non-cumulative changes.
- You are not able to reload marker data from the archive directly into the marker of a non-cumulative InfoCube. This is because the archived data is accessed by the export DataSource of the original InfoCube, which does not support non-cumulative initialization. We recommend the following temporary solution: You can use a full update to load the marker data as non-cumulative changes if you change the value of the time characteristic to a value that lies in the distant past (before the likely value of a lower threshold of the validity area) and if the request is compressed adjusted to the marker. The point at which the data is reloaded should be hidden by adjusting the validity area.

  The validity area is not adjusted automatically if you delete data selectively (particularly in connection with an archiving run.) You need to use the transaction RSDV to adjust the validity area manually wherever necessary.
- If the archiving run contains a mixture of non-cumulative changes and marker data, you can set a selection condition for the *record type* characteristic (0RECORDTP) during the extraction of the data from the archive into the InfoPackage to keep the non-cumulative changes and the marker data separate. 0RECORDTP = ’0’ indicates the non-cumulative changes and 0RECORDTP = ’1’ indicates the value of the marker that is stored as a non-cumulative change (= end non-cumulative.)
- You must compress non-cumulative changes immediately after you have loaded them from the archive, **without** adjusting the marker.

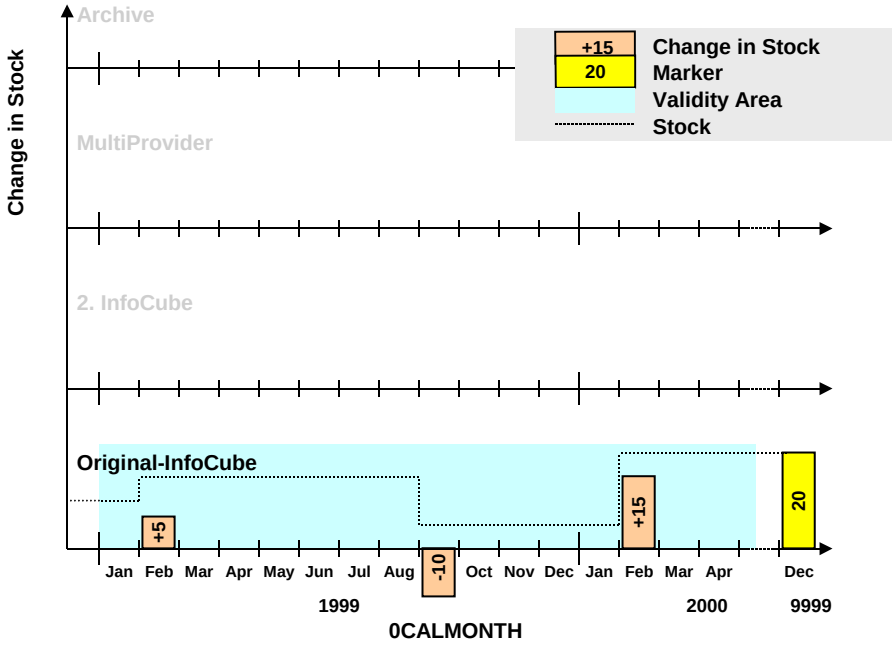
See also:

Example of Archiving Data from Non-Cumulative InfoCubes [Page 763]

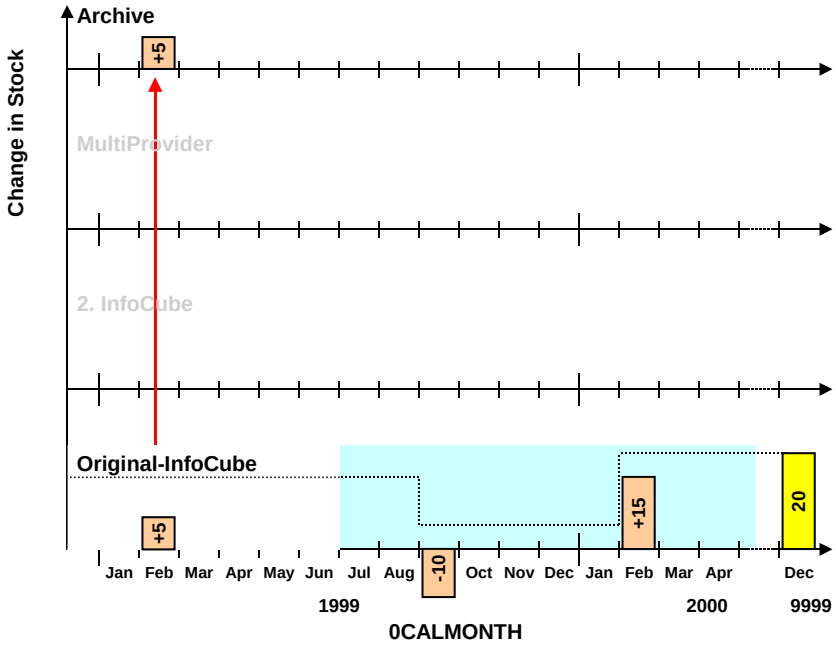
4.4.9.2.4.2 Example of Archiving Data from Non-Cumulative InfoCubes

The following example shows how you archive non-cumulative changes as time slots, how you reload data into a copy of the original InfoCube from which the data came, and how you prepare all the data so that it can be used in reports using a MultiProvider. You have to maintain the validity area (marked in blue) manually. You use transaction RSDV to do this.

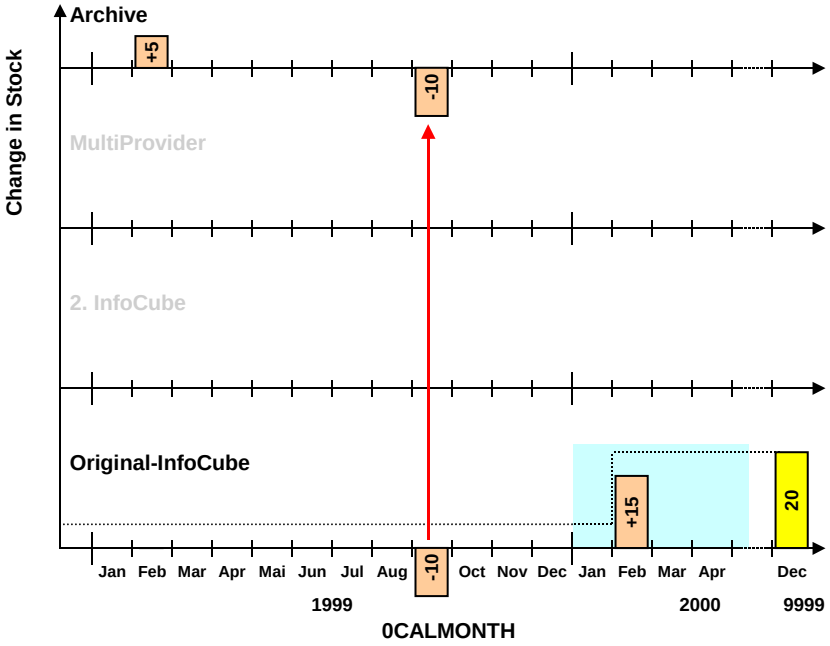
This is the situation before archiving: The InfoCube is filled with data on in-comings and out-goings.



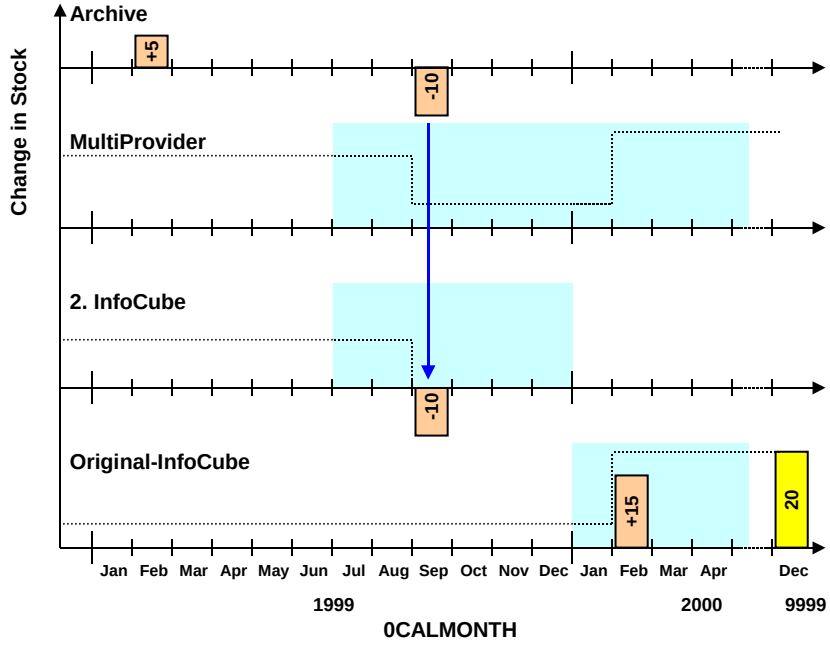
Die nächste Grafik zeigt den Archivierungslauf für die erste Zeitscheibe (Januar bis Juni 1999):



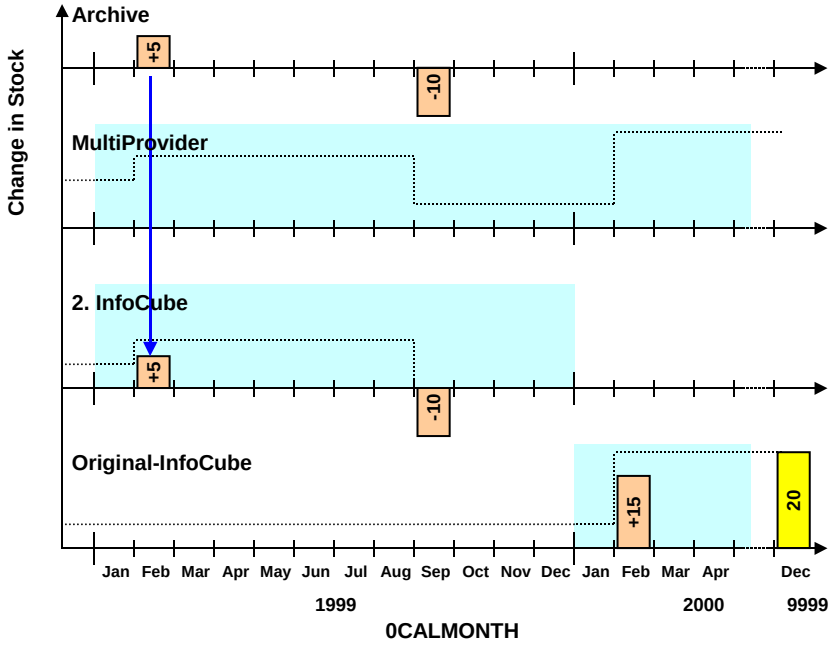
The next graphic shows the archiving run for the second time slot (July to December 1999.)



The next step is to load the data from the second time slot (July to December 1999) into the copy of the InfoCube.



The next step is to load the data from the first time slot (January to June 1999) into the copy of the InfoCube.



When you run reports using the MultiProvider, all the data, including the archived data, is available.

4.4.9.2.5 Checking an Archive File

Use

You can use the Extractor Checker (transaction RSA3) to check an archive file.

Procedure

1. Call up transaction RSA3.
2. Select the DataSource for your archived Object. The name of the DataSource begins with an 8, followed by the technical name of the DataStore object or InfoCube.
3. Choose the option *Archive Only* to extract the archived data.
4. Select your archive file using the *Archive File Selection* pushbutton.
5. Start the extraction.

4.4.9.2.6 Reloading Data from Archive Files

Prerequisites

SAP does not recommend that you reload directly into the original object. Instead, you should create an InfoProvider with the same (or a similar) structure, with transformation rules between your archived InfoProvider and this new InfoProvider.

Exception: With DataStore objects, you can switch on the protection function for the archived area. This enables you to reload data out of the archive administration into the active data table for the DataStore object. Also refer to the *Time Slots* section in Archiving Object [Page 755].

In order to reload archived files, you need the export DataSource of the archived InfoProvider.

Procedure for DataStore Objects

1. To reload archived data into a DataStore object of the same structure, in the InfoProvider tree, from the context menu of the archived DataStore object, choose *Update DataStore Data into InfoProviders.*
2. Choose *Full Update* as the update mode (only this is supported for archive data) and choose *Update*. The Scheduler appears.
3. From the *Data Selection* tab page in the Scheduler, choose your archived file using the *Archive Selection* pushbutton.
4. Start the load process.

Procedure for InfoCubes

5. In order to reload archived data into an InfoCube with the same structure, in the InfoProvider tree, from the context menu of the archived InfoCube, choose *Generate Export DataSource*. You will receive a success message when the Export DataSource was generated successfully.
6. Go to the InfoSource tree. Using the main menu, choose *Settings* → *Display the Generated Objects.* Choose *Show Generated Objects* and confirm.
7. Find the generated InfoSource 8<InfoCubeName> for your InfoCube.
8. Create an InfoPackage for this InfoPackage.
9. From the *Data Selection* tab page in the Scheduler, choose your archived file using the *Archive Selection* pushbutton.
10. On the *Update* tab page, select *Full Update* as the update mode.
11. Start the load process.

Result

You have loaded the archived data into a new InfoProvider and can activate it to create reports with this data.

You can also combine both of your InfoProviders in one MultiProvider to merge the non-archived data with the archived data that you reloaded.

Note: When reloading data into the original object, an inconsistent delta determination may arise for data targets that are subsequently stored. There is also the danger that data is archived more than once.

4.5 Applying Extensibility

Purpose

With the SAP enhancement concept, you can make enhancements to the standard in the BI system. Within the BI system, you use function exits and BAdIs to make these enhancements.

For more information, see the documentation for the Application Server under Changing the SAP Standard (BC) [External], in particular the sections on Customer Exits [External] and Business Add-Ins [External].

For more information about the enhancement options in the BI system, see Customizing under *SAP Customizing Implementation Guide* → *SAP NetWeaver* → *Business Intelligence* → *BI Enhancements*.

Enhancing DataSources

SAP enhancement RSAP0001 and Business Add-In RSU5_SAPI_BADI are available for enhancing BI DataSources.

For more information, see Editing DataSources and Application Component Hierarchies [Page 550].

Enhancements in Transformation Library and Formula Editor

You use Business Add-In RSAR_CONNECTOR to integrate functions that you have defined into the transformation library of the formula editor. In doing so, you can also make special functions available for frequent use that are not contained in the transformation library.

For more information, see BAdI: Customer-Defined Functions in the Formula Editor [Page 237].

4.6 Enabling Globalization

Purpose

You can translate the short texts and long texts for BI objects.

For more information, see Translating Text for BI Objects [Page 768].

4.6.1 Translating Text for BI Objects

Use

To translate the short and long texts of BI metadata objects, you use special functions of the Data Warehousing Workbench in your BI system as well as the translation environment provided by the SAP Web Application Server (ABAP).

Procedure

Do the following:

Step	Description
1	Collect the BI metadata objects to be translated along with all the dependent objects in the functional area *Translation* of the Data Warehousing Workbench.
2	Create a piece list with the objects to be translated in the functional area *Translation* of the Data Warehousing Workbench. The standard case is that a piece list is created. However, since a piece list **cannot** be created in a system that has the global setting “cannot be changed”, you use a workbench request in this case. For more information, see SAP Note 885571.
3	Generate an object list in the translation environment.
4	Process the worklist in the translation environment.
5	Create the M version of the BI objects using program RSO_AFTER_TRANSLATION in the ABAP editor.

You can then transport the translated objects to other systems.

Proceed as follows:

1. Collecting Objects for Translation in Data Warehousing Workbench

1. Start the **Data Warehousing Workbench**. (One way to do this is to choose *Modeling* → *Data Warehousing Workbench*: *Modeling* on the SAP Easy Access screen.) In the navigation window or the *Goto* menu, choose the *Translation* functional area.

   The procedure for steps 2 to 5 is very much the same as the procedure for installing Business Content (steps 2 to 5). For more information, see Installing BI Content [Page 435] .

2. If you want to assign a source system, select *Source System Assignment*. The *Choose Source System by Default* dialog box appears.

3. Make your settings in the selection lists *Grouping* and *Collection Mode.*

4. In the navigation window of the Data Warehousing Workbench, select *Object Types*. In the central area of the screen, the system shows all the object types in the BI system that can be translated.

   You can use *Grouping* to include all the objects before or after the selected object in the data flow in the translation. This allows you to translate a structured source system scenario, for example, from InfoSources and InfoCubes through to queries and workbooks.

5. In the central area of the screen, choose *Select Objects* to select the required object type of the object(s) that you want to translate. Use drag and drop to transfer them to *Collected Objects* in the right-hand area of the screen.

6. In the *Collected Objects* area of the screen, the system displays the selected objects and all their dependent objects. The worklist automatically includes the following objects for object translation:

Worklist for an Object

Object	Example
BI object	Query
Sub-objects for this object	For a query: InfoObjects (characteristics and key figures) and the predefined structures of an InfoProvider.
Objects connected to this object	For a query: those workbooks in which the query is embedded.

Note the following features:

- The system only writes the selected objects to the piece list for the translation.
- Objects in the SAP namespace (beginning with digit 0) are automatically **not** selected.
- Like the *Transport Connection* functional area, in the context menu you can select everything for a node or delete selections.
- Texts are only translated if they belong to an object that is in the active version (A) or in a Business Content development system in the delivery version (D) because only these objects are transported. If you have changed the object texts, you must activate these objects **before translation.**
- **Navigation attributes** have to be translated in addition to InfoObjects if their texts are not copied from the attribute characteristic. For the navigation attribute 0COSTCENTER__0RESP_PERS, 0RESP_PERS is not translated; characteristic 0COSTCENTER is translated with the text of this navigation attribute. Execute program RADNTLANG if the texts of the navigation attribute are not provided for translation. For more information, see SAP Note 480671.
- The texts of the **DDIC objects** generated for the metadata objects are not displayed on the interface in the BI system. Therefore they do not need to be translated.

7. Choose *Translate.* The *Piece List for Translation* screen appears.

2. Generating a Piece List in the Data Warehousing Workbench

A piece list containing the objects to be translated is required to generate a worklist.

1. Choose *Create.* The *Select Request Type* dialog box appears.
2. Choose *Piece List.* The *Request* dialog box appears.

   If you cannot use a piece list because, for example, the system cannot be changed, you can use a workbench request instead.
3. Enter a name (the name has to be in the customer namespace [start with Y or Z]) and a short description for the piece list and save your entries. The *Object Editing: Initial Screen* dialog box appears.
4. Choose *Continue*. The *Create Object Catalog Entry* dialog box appears.
5. Enter a package and save your entries.
6. Choose *Continue*. The *Piece List for Translation* dialog box appears again. You have successfully created a piece list for translation.

7. Choose *Continue*. The *Translation Environment* screen appears (transaction code LXE_MASTER).

3. Generate an object list in the translation environment.

In order to generate a worklist in the translation environment, you first have to create an object list for the piece list in the **translation environment (transaction code LXE_MASTER)**.

1. On the *Evaluations* tab page, select *Object Lists*. The *Object Lists* dialog box appears.
2. You can create a new object list with *Create*.

   More information: Creating Object Lists [External] in section *Maintain parameters for object list according to piece list (ABAP developments only)*.

   a. Enter a text in the field *Description Text* and choose *Save*. The *Parameters* dialog box appears. The number and description of the selected object list is displayed in the upper half of the screen.

   b. In the *ABAP System* pane next to the *Transports* field, select *No Entries* and choose *Continue*. The *Transports/Piece Lists* dialog box appears. The number of the selected object list is displayed in the upper half of the screen.

   c. Enter the name of the piece list you want to include in the object list. If you want to evaluate more than one piece list, you must create an entry for each piece list.

   d. In the next two columns, enter * as the transport type and transport status; do **not** enter a value in the other fields (date and time).

   e. Save your entries for the piece list. The *Parameters* dialog box appears. The number of selected piece lists is given on the pushbutton next to the field *Transports*.

   f. Save your entries for the parameters of the object list.
3. Generate the required object list by selecting it and choosing *Activate*. The *Start Time* dialog box appears and you can schedule the background job. (The name of the job is OBJLIST_xxxx, where xxxx is the number of the object list.)

   More information: Creating Object Lists [External] in section *Generate object list*

4. Process the worklist in the translation environment.

Once the job for generating the object list has ended, you can translate the objects contained in this list from a worklist in the **translation environment (transaction code SE63)**. Only the simplest case of such a translation is described below.

More information: Translation Tools for Translators (BC-DOC-TTL) [External], especially the sections about Personal Worklists [External].

1. Check your default settings.

   a. Choose *Utilities* → *Default Settings*.

b. From the input help, select the version of the source and target language (for example "deDE" for German and "enUS" for English US).

2. Call up your personal worklist with *Worklist* → *Standard*. The *My Worklist: Parameters* dialog box appears.

   a. Define your parameters (source language, target language, worklist number).

      If you want to reset an existing personal worklist and/or a reservation number for calling a personal worklist, select the *Reset Worklist/Reservation* checkbox.

   b. Choose *Copy.*

   The *Get Worklist* screen appears.

3. Choose the *Object List* tab page.

   a. In the field *Object List*, select the object list from which you want to call the objects in your personal worklist.

   b. Select *Deactivate Selection.*

   c. Choose *Load Objects*.

      More information: Accessing a Personal Worklist [External] in section *Accessing a personal worklist using an object list*.

   The *Worklist* screen appears containing the objects of your object list that are to be translated.

4. Enter a text in the target language for the objects to be translated, save your entries, and define the standard quality status "S" by choosing .

   More information: Translating Objects via a Worklist [External].

5. Once you have completed translation, save the entire worklist.

5. Postprocessing Using Program in ABAP Editor

Texts are only translated for objects in the A or D version. Texts of objects in the modified version (M) are **not** translated. Therefore, when you call the maintenance transactions in the target language, the system displays the old texts (if available) for these objects first.

A program is available for viewing the newly translated texts in the system, also for objects in the modified version (M).

1. Call the **ABAP Editor (transaction SE38)**.
2. Execute program RSO_AFTER_TRANSLATION with the following parameters:

   Request/Task: Name of piece list

   Language Key: Source language

   Language Key: Target language

   The system copies the texts for the objects in the piece list.

Result

The translated texts are available in your BI system.

Transporting the Translated Texts in the Data Warehousing Workbench

If you want to make the new texts available in other systems, you can transport them in the *Transport Connection* functional area of the Data Warehousing Workbench:

1. Collect the translated objects.
2. To transport the translated objects, choose *Transport All Below* from the context menu. This writes all the collected objects explicitly to a request. This is necessary if some of the objects have already been transported and therefore are not selected by default.

   During the transport the entire objects (not just the texts) are transported. Depending on the number of objects, the import may take some time.

   More information: Transport System [External]

5 Ensuring Quality

This section deals with the quality measures involved in development such as testing, logging and tracing, and debugging.

5.1 Testing

Purpose

BI Content Analyzer

The BI Content Analyzer provides a detailed analysis of BI Content objects. The check programs identify these objects with respect to errors and inconsistencies.

For more information, see BI Content Analyzer [Page 774].

Analysis and Repair Environment

In the analysis and repair environment, you can perform consistency checks on the data and metadata stored in a BI system. This predominantly tests the foreign key relationships between the individual tables of the enhanced star schema of the BI system.

For more information, see Analysis and Repair Environment [Page 792].

Extraction Test

Independent of the BI system, you can test extraction from DataSources that were created using generic DataSource maintenance.

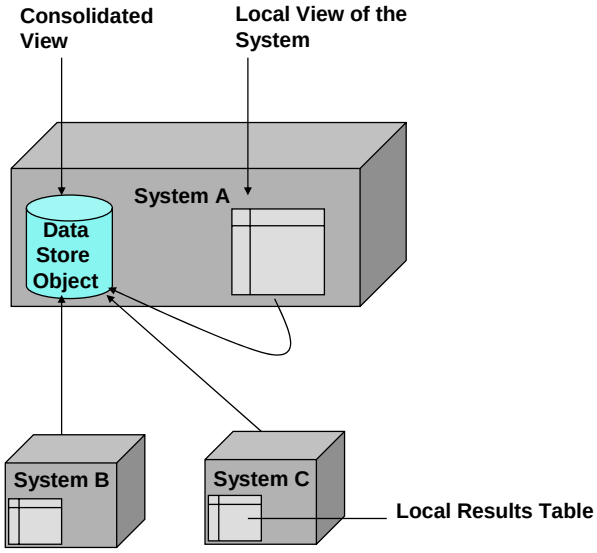
For more information, see Testing Extraction [Page 547].

5.1.1 BI Content Analyzer

Purpose

The BI Content Analyzer serves to provide a detailed analysis of BI Content objects. The objects were previously identified by check programs, with respect to errors and inconsistencies. The results of these checks can be viewed under different aspects in order to easily keep problems under control. There are two scenarios:

1. You analyze your data locally for a respective BI system. Here you can access the results of the check program (saved in transparent tables in the BI system) directly.
2. You analyze your data in consolidated form from a central BW system. To do this, load the results of the check program from the transparent tables into the InfoProviders (DataStore objects) of the central BI system you chose using extractors. This loaded data is then available in the BI Content Analyzer (transaction RSBICA) in the filter selection *BI Systems*.



Features

- You can schedule check programs to analyze your BI systems based on their data. You can view the results of all the check programs included in the results area of the BI Content Analyzer.
- The BI Content Analyzer only shows erroneous objects.
- You can either analyze the results immediately in the BI system or by using the BI reporting functions. The BI Content Analyzer itself provides extensive reporting options. For many objects, the BI Content Analyzer can provide a detailed error message in addition to identifying the error.
- You can also load the results of the check program into other BI systems. A DataStore object provides a central entry point for the analysis of your BI system landscape.

The Main Screen Has Three Areas:

1. The **monitor area** refers to the results area (see point 3). You see an aggregation of results that depends on the filters set in the filter area:
   a. The totals of the objects determined by the BI Content Analyzer and displayed in the results area according to the current selection in the *Priority* filter. You can find this information in the fields *Priority 1*, *Priority 2*, *Priority 3* and *Priority 4.*
   b. The time stamp of the object in the results area that was the first to be determined by the check programs. You can find this information in the two fields for *Last Update*.
2. In the **filter area** you can restrict the values of the results area for the results area. This is done using the selection of display for the
   - *BI system.* You can choose between
     i. the BI system set as local

        or (if check data from other BI system has already been loaded):
     ii. individual BI systems connected using the DataStore object
     iii. all BI systems connected using the DataStore object including the BI system that is set as local
   - *Transport Filter*: You can also analyze through activation of the transport filter and determination of individual transports of its objects. However, if these objects are not part of the results table, they are displayed under the special check category *No Check* in the results area. Under *No Check* you will see all the objects that were either non-BI objects or were not included in the check programs due to the restrictions made in the global defaults (see Setting Global Defaults [Page 781]).

     The transport filter only includes the local BI system.
   - *Check Category* You can decide between one of the provided check categories or all of them (*all checks)*. For more information, see BI Content Analyzer Checks [Page 787].
   - *Priority* You can choose between
     i. Very high (1)
     ii. High (2)
     iii. Medium (3)
     iv. Low (4)
     v. All problems
     vi. Exceptions longer than four weeks. See Setting Exceptions [Page 784].
   - *Grouping* Here you determine the arrangement of the elements in the drilldown of the tree structure. You can choose between the sequences
     i. Package / Check / Type / Object
     ii. Type / Check / Object
     iii. Author / Check / Type / Object
3. In the **results area** you find the tree hierarchy set up using the *Grouping* filter in addition to a tabular list of all of the objects that were checked and collected by the BI Content Analyzer.

- You can expand the tree hierarchy up to the object entry and view all of the important details in the table for the object displayed.
- When you double click on the row entry you get the error log for the object displayed with short texts. When you double click on the short text, the long text is displayed. It contains repair instructions.

In the Main Menu:

- *Edit → Update Exceptions* Subsequent changes to the settings for the exceptions can be compared with the results area immediately. You do not have to schedule the check program again.
- *Edit → Update Priority* Subsequent changes to the settings for prioritizing the various check categories can be compared with the results area immediately.

  You do not have to schedule the check program again for this update.
- *Edit → Start Check Reports*. For more information, see Scheduling Check Programs [Page 786].
- *Analysis → Update* You can update the display in the results area here.
- *Delete → Status Table* Here you can delete the entire results table from the check programs.
- *Delete → Message Table* Instead of deleting the entire results table, you can only delete the messages for all check results that are the basis of the error logs.
- *Settings* For more information, see Important Defaults [Page 781].

5.1.1.1 Implementing Business Content for the BI Content Analyzer

Purpose

Business Content and ABAP code are delivered for the BI Content Analyzer as the transport. You can use Business Content to load the results of the check programs (ABAP code) from other BI systems into one of the central BI systems you defined. Then you have the option of analyzing all of your BI systems at a glance.

The transport has to be installed into every BI system from which you wish to extract analysis data. Have your system administrator install the transport.

Process Flow

In the Source Systems:

1. Install the following DataSources from Business Content in all systems from which you wish to extract.

   0RS_CHECKCAT_TEXT

   0RS_MSGODS_DATASOURCE

   0RS_OBJTYP_TEXT_UPLOAD

   0RS_TRANDATA_UPLOAD

Ignore the warnings about missing application components here. For more information on the procedure, see the SAP Library under *SAP NetWeaver → Information Integration → SAP NetWeaver Business Intelligence → Data Warehousing → Data Retrieval → Data Extraction from SAP Source Systems → Customizing the Extractors → Installing Business Content DataSources.*

2. In transaction RSBICA, make the necessary settings for the check programs and execute the programs.

In the BW System into Which the Data Is to Be Loaded:

3. Replicate the metadata into the BI system into which you want to load the data from the check programs. For more information on the procedure, see the SAP Library under *SAP NetWeaver → Information Integration → SAP NetWeaver Business Intelligence → Data Warehousing → Data Retrieval → Data Extraction from SAP Source Systems → Metadata Upload for SAP Systems.*
4. Install the Business Content for the BI Content Analyzer. Collect and activate the required objects for:
   - the DataStore object 0RS_RPODS: Choose all the relevant systems (Myself as well) and choose *Data Flow Before → Collect → Activate*.
   - the DataStore object 0RS_RPODS: Choose the *Data Flow After → Collect → Activate*.
   - the DataStore object 0RS_MSODS: Choose the *Data Flow Before → Collect → Activate*.

   For more information, see the SAP Library under *SAP NetWeaver → Information Integration → SAP NetWeaver Business Intelligence → Data Warehousing → Data Warehouse Management → Business Content (Versions) → Install Business Content*.
5. Load the texts for the InfoObject 0RS_CCAT. These are the texts for the check categories. They have to be loaded once to make them available as master data.
6. Load the transaction data using InfoSources 0RS_TRAN_INFPACK and 0RS_MESSAGE_INFOSOURCE.
7. Activate the data in the DataStore objects 0RS_RPODS and 0RS_MSGOD. For more information on the procedure, see the SAP Library under *SAP NetWeaver → Information Integration → SAP NetWeaver Business Intelligence → Data Warehousing → Modeling → Data Target → DataStore Object → Activate Data in the DataStore Object.*
8. Update the data from the DataStore object 0RS_RPODS to the InfoObject 0RS_OBJNAME. For more information on the procedure, see the SAP Library under *SAP NetWeaver → Information Integration → SAP NetWeaver Business Intelligence → Data Warehousing → Modeling → Data Target → DataStore Object → Update DataStore Object Data.*
9. Execute the attribute change run for the InfoObject 0RS_OBJNAME. For more information on the procedure, see the SAP Library under *SAP NetWeaver → Information Integration → SAP NetWeaver Business Intelligence → BI Platform → OLAP → Performance Optimization → Aggregate → Applying Hierarchy and Attribute Changes.*
10. Activate the Report-Report Interface for the report RSCONTENT_RRI_MESSAGE_POPUP. Start the transaction RSBBS and make the following receiver-sender assignment for the query 0RS_ENHANCEMENT.
    - Report type: ABAP/4 report

- Target system: local
- Report: RSCONTENT_RRI_MESSAGE_POPUP

Save the entry.

11. Call the Web template 0RS_BICA.

5.1.1.1.1 Roles

5.1.1.1.1.1 Content Administrator

Role: SAP_BW_CONTENT_ADMIN

This role delivers central access to the BI Content Analyzer. The BI Content Analyzer allows you to perform detailed analysis of BI Content objects that have been identified by check programs as containing errors or being inconsistent. You use the role to analyze and evaluate customer and partner development in SAP NetWeaver BI.

5.1.1.1.2 Web Templates

5.1.1.1.2.1 Web Template: BI Cont Analyser_Reporting

WebTemplate: 0RS_BICA

This Web template returns the results of the Content Analyzer. You can also view these results in the system in the Content Analyzer (transaction: RSBICA). It is based on queries ORS_ENHANCEMENT and ORS_ENHANCEMENT_GRAPH.

This Web template provides a user-friendly interface, including filter options that allow you to display results.

To activate the Report-Report-Interface, make the following sender/recipient assignment in the BI system in transaction RSBBS:

- Report type: ABAP/4 Report
- Target system: Local
- Report: RSCONTENT_RRI_MESSAGE_POPUP

Prerequisites

Copy Web template 0RS_BICA to the Business Content (transaction: RSOR).

Data Modeling

Requires

Object Type	Name	Technical Name
Query	Reporting Enhancement_Graph	9PZLKABQ7YRDN01BLT1ND2WTR
Query	Reporting Enhancement	BNMOBK4D6C1UUZUKFTT13YPM6

5.1.1.1.3 Queries

5.1.1.1.3.1 Reporting Enhancement

Query: 0RS_ENHANCEMENT

You use this query to display the results of the Content Analyzer. The query provides the same functions as the Content Analyzer in the application.

Structure

Filters

Object Type	Name	Technical Name	Value Restriction / Calculation
	Analyzer — Exception	0RS_EXCEPT	= X [Exclude from Selection]

Free Characteristics

Object Type	Name	Technical Name
	Analyzer — Development Class	0RS_DEVCLASS
	Analyzer — Priority	0RS_PRIORITY
	Analyzer — Check Type	0RS_CHECKTY
	Analyzer — System	0RS_SYSTYPE
	Analyzer — Author	0RS_AUTHOR

Rows

Object Type	Name	Technical Name	Value Restriction / Calculation
	Analyzer — Object Type	0RS_OBJTYPE	—
	Analyzer — Logical System	0RS_LOGSYS	—
	Analyzer — Check Category	0RS_CCAT	—

Rows

Object Type	Name	Technical Name	Value Restriction / Calculation
Analyzer — Object Name		0RS_OBJNAME	—

Columns

Object Type	Name	Technical Name	Value Restriction / Calculation
Analyzer — Error Count		0RS_COUNT	—

Report/Report Interfaces

Object Type	Name	Technical Name
RSCONTENT_RRI_MESSAGE_POPUP		RSCONTENT_RRI_MESSAGE_POPUP

5.1.1.1.3.2 Reporting Enhancement_Graph

Query: 0RS_ENHANCEMENT_GRAPH

You use this query to display the results of the Content Analyzer in a bar chart, sorted according to check category.

Structure

Free Characteristics

Object Type	Name	Technical Name
Analyzer — Development Class		0RS_DEVCLAS

Rows

Object Type	Name	Technical Name	Value Restriction / Calculation
Analyzer — Check Category		0RS_CCAT	—

Columns

Object Type	Name	Technical Name	Value Restriction / Calculation
Analyzer Error Count		0RS_COUNT	—

Report/Report Interfaces

Object Type	Name	Technical Name
RSCONTENT_RRI_MESSAGE_POPUP		RSCONTENT_RRI_MESSAGE_POPUP

5.1.1.2 Important Settings

Process Flow

In order to be able to analyze erroneous objects with the BI Connect Analyzer, you first have to make a number of settings after starting transaction RSBICA via *Settings*.

1. *Global Settings:* Here you change the analysis behavior of the BI Content Analyzer and the prestored check programs. For more information, see Setting Global Settings [Page 781].
2. *Check Execution*: Here you specify the check types for use and group them as is required for the scheduling of check programs that follows. For more information see Determining Check Execution [Page 783].
3. *Prioritization*: Here you set the error priority for each check category. For more information, see Setting Priorities [Page 784].
4. *Exceptions*: Here you set exceptions for the display of existing objects in the results area. For more information, see Setting Exceptions [Page 784].
5. *Special Objects*: Here you set source-system dependent objects for auto activation. For more information, see Including Source-System Dependent Objects [Page 785].
6. *Packages* Here you release packages for the check programs. For more information, see Releasing Packages [Page 786].

Result

By making these basic settings, during analysis, you will receive exactly the data displayed that is interesting for you. Next you can schedule the check programs for the set BI systems using *Edit* → *Set Check Reports* in the main menu and start the error analysis. For more information, see Scheduling Check Programs [Page 786].

5.1.1.2.1 Setting Global Settings

Procedure

1. On the main menu, choose *Settings* → *Global Settings.*

   The *Global Settings for BI Content Analyzer* view appears.

   You can only make new entries and settings in change mode in the detail view.

   The global settings must be set individually for **each** participating BI system.
2. Set the relevant logical BI system for the check programs and the BI Content Analyzer.
3. Under *Settings: BI Content Check*, set **Details with Relevance for the Check Programs.**
   - *Namespace*: By entering a specific namespace, you make a restriction. If you make no entry, all objects are included in the check program.

Objects in the *Customer Namespace* start with letters from A to Z. If you only want to include objects in the A namespace, enter **A**.

Objects in the *Partner Namespace* include their own registered namespaces, such as **/ABC/**.

The use of wildcards (*****) is not allowed in the BI Content Analyzer.

- *Customer Namespace*: If you want to include objects from the customer naming convention in the check programs, activate this field.
- *Partner Namespace*: If you want to include objects from the partner naming convention (registered namespace) in the check programs, activate this field.

Activate at least one of the two: *Customer Namespace* or *Partner Namespace*. Otherwise **no** objects will be included.

- *Package:* If you want to release packages (development classes) you determine yourself for the check programs, activate this field. You determine which packages these should be in the main menu: *Settings* → *Packages*. See Releasing Packages [Page 786].
- *XAND Connection* (between namespace and package): One OR connection is set by default. To create an AND connection, activate this field.

4. Under *Settings: BI Content Analyzer*, set **Details with Relevance for the BI Content Analyzer** with the objects that were already checked.
   - *Use DataStore* (for reporting): All BI systems that are available in the DataStore object of the BI Content Analyzer (0RS_RPODS) are included in the analysis.

     This field is disabled by default.

     Performance while the tree structure is being built in the results are is better when the setting *Use DataStore* is inactive.
   - *Hide Exceptions*: You can set the exceptions for check objects. These types of objects are not displayed in the results area. You determine which objects these should be in the main menu: *Settings* → *Exceptions*. See Setting Exceptions [Page 784].

     This field is enabled by default.
   - *Order/Task*: Number of order or task.
   - *Max. No. of Lines*: Maximum number of lines that can be displayed in the results area. The recommended value is 15,000.

     If more rows are available than has been set, a warning is displayed between the filter and results area that rows have been cut off. In this case you should decrease the size of the results area using additional filter restrictions.
5. Repeat steps 2 to 4 if you want to set entries for further BI systems that you then wish to transport into the system landscape.

5.1.1.2.2 Determining Check Execution

Use

Here you specify the check types for use and group them as is required for the scheduling of check programs that follows.

Here only the settings that are actually required for later work by the check programs are made. You schedule the check programs more concretely using the main menu, *Edit → Start Check Reports*. For more information, see Scheduling Check Programs [Page 786].

Procedure

1. On the main menu, choose *Settings → Check Execution*.

   The *Customizing View for BI Content Analyzer Checks* view appears.

   You can only make new entries and settings in change mode in the detail view.

2. In the *Logical System* column, enter the BI system that is to be checked by the check programs.
3. In the *Check Type* column, choose a check type for the BI system entered under 2. Here you determine the errors the check program is to search for. For more information on check types and check categories, see BI Content Analyzer Checks [Page 787].
4. In the *Job Name* column, enter a free selectable job name for the check type selected under 3. You can enter more checks later by repeating the individual steps. As needed, you can assign identical job names for the individual check types in order to group the check types. These types of grouped check types are handled simultaneously by the check programs at a later time. See also Scheduling Check Programs [Page 786].
5. Use the fields in the *Execute* column to determine whether you actually want to run a check.
6. The fields in the column *Execute Simultaneously* are inactive by default; checks are usually run one after the other. The setting *Execute Simultaneously* is relevant for some check types that can be executed as *Multi Check*. These check types include:
   - Object status
   - Object collection error
   - Inconsistent naming convention

   If you want to run *Multi Checks*, activate *Execute Simultaneously*.

   For more information on the individual check types, see BI Content Analyzer Checks [Page 787].
7. In the *Version* column, set the object version that is to be included in the check programs. The following versions are permitted:
   - Active (A version)
   - Content (D version)
8. Repeat steps 2 to 7 when you specify more check types and want to group them.

5.1.1.2.3 Setting Priorities

Use

You assign a priority level to check categories with priority maintenance. On the main screen, you can display the errors during the analysis according to these types of priority groups (filter *Priority*) in the results area.

Procedure

1. On the main menu, choose *Settings* → *Prioritization*.

   The *Priority Maintenance for Results for BI Content Analyzer Checks* view appears.

   You can only make new entries and settings in change mode in the detail view.

2. Set priorities for individual check categories along the sequence of the key
   - *Object Type* (TLOGO)
   - *System Type*
     i. Development system
     ii. Test system

     In the BI Content Analyzer, a system is a test system when the *Content System* field is inactive on the RSADMINSV view.

   You can choose from the following levels:
   - Very high (1)
   - High (2)
   - Medium (3)
   - Low (4)
   - All problems
   - Exceptions longer than four weeks. See Setting Exceptions [Page 784].

   Here you override the BW default priority levels for the check categories. You will find a list of these priority levels at BI Content Analyzer Checks [Page 787].

3. Repeat step 2 if you want to assign your own priority levels to further check categories.

5.1.1.2.4 Setting Exceptions

Use

Here you set exceptions for displaying erroneous objects that are not relevant for your analysis. Objects that you have classified as an exception are not displayed in the results area for further analysis once the test program has been run successfully.

Procedure

1. On the main menu, choose *Settings* → *Exceptions*.

The *View Maintenance of BI Content Checks* view appears.

You can only make new entries and settings in change mode in the detail view.

All settings you make here only take effect when the field *Hide Exceptions* is active under *Settings* → *Global Defaults*.

2. Assign the following keys to the required entry:
   - *Object type* (TLOGO)
   - *Object name*
   - *Category* (check category)
3. Using the *Checked* and *Exception* column to designate the object that is to be treated as an exception by the BI Content Analyzer from now on. To do this, activate one of the two columns.
   - *Checked*: The error is known and will be rectified soon. The selected object will only be displayed in the results area of the BI Content Analyzer once four weeks have passed since the last time the check program was run, as long as the error still exists.
   - *Exception*: general exception. The selected object is no longer displayed in the results area.
4. Compile a note about how to handle the object in the *Message or Comment* column.
5. Repeat steps 2 to 4 if you wish to treat other objects as exceptions.

5.1.1.2.5 Including Source-System Dependent Objects

Use

Here you can define source-system dependent objects that are to be checked later by check programs.

Procedure

1. On the main menu, choose *Settings* → *Special Objects*.

   The *Source-System Dependent Objects for BI Content Analyzer* screen is displayed.

   You can only make new entries and settings in change mode in the detail view.
2. Assign the following keys to the required entry:
   - *Logical System*
   - *Object Type* (TLOGO)
   - *Object Name*
3. In the *Responsible Person* column, specify a person responsible for the entry.
4. In the *Name of Department* column, specify the affected department.
5. Repeat steps 2 to 4 if you wish to include other objects.

5.1.1.2.6 Releasing Packages

Use

For some objects, restriction using the naming conventions is not sufficient for having them included in the check programs. When you specifically release individual packages for the check programs, you can check objects from specific development areas.

Procedure

1. On the main menu, choose *Settings* → *Packages*.

   The *Packages for BI Content* overview appears.

   You can only make new entries in change mode in the detail view.

   All settings you make here only take effect when the field *Package* is active under *Settings* → *Global Settings*.

2. Enter the package of your choice.
3. Repeat step 2 if you want to release more packages.

5.1.1.3 Scheduling Check Programs

Use

You choose an execution time for the check programs. The (possibly grouped) check types that are connected to a job type are executed.

As an alternative you can also schedule and start the report RSCONTENT_ANALYSIS_FRAME_START.

Prerequisites

You must have already made the associated settings. See also: Important Settings [Page 781].

Procedure

1. On the main menu, choose *Settings* → *Start Check Reports*

   The *Customizing View for BI Content Analyzer Checks* view is displayed.

2. First make general settings in the following fields
   - *Logical System:* Enter the BI system in which the job is to run.
   - *Job Name*: Select a job name. In the input help (F4), the jobs are entered that you had determined in the settings (see Determining Check Execution [Page 783]).

- *RFC Destination:* If you want to use a connected system for the check programs other than the one entered into the *Logical System* field, enter the name of the associated RFC destination of the system. You will find the name of the system connected to the RFC destinations in the value help (F4).

  However, only those RFC destinations are allowed that were once created as a dialog RFC connection. The RFC user has to be a *Dialog User*. Otherwise there will be an error message.

  If you want to include other systems using their RFC destinations, their priorities for check categories are important to the check programs, not those of the logical system (see Setting Priorities [Page 784]) for which you have made the settings. You can either transport these settings into the included target systems or maintain them manually.

3. Then define the following in the *Start Time Values*:
   - *Immediately*
   - *Date/Time:* Enter the exact or latest execution date.
   - *By Job*: Enter the name of the job that has to be finished before the background control processes your job.
   - *By Event:*
     - *Event:* Enter the event name with which the job is to be connected. You can specify the predefined events for SAP applications or SAP system kernels or you can specify events that were defined manually. You can find out which SAP events you can use by looking in the documentation for SAP applications.
     - *Parameter:* If an event requires a parameter, enter the parameter in this field. To determine whether an event outputs a parameter, look for the appropriate SAP application in the documentation.
   - *Execute Job Periodically:* Activate this field to schedule the job whenever an event occurs.

5.1.1.4 BI Content Analyzer Checks

Use

The check programs determine errors and inconsistencies of the BI Content objects of the BI system. The selection according to which routines the selected objects are handled are made when you assign **check types** (*Settings* → *Check Execution*).

Check types can be grouped by assigning identical job names. Check types with identical job name are executed simultaneously. Some check types can be executed as a *Multi Check*. Check types have a two-digit ID.

Each check type has at least one subordinate **check category**. Check categories are used in the BI Content Analyzer to filter the check results in the results area. Check categories have a three-digit ID.

Features

The following tables provide an overview of all check types and their categories.

Type: Object Status- Check (Multi)

Type ID	Assigned Check Categories	Categ. ID	Standard Priority
UNIT	*Object Status: A Version Newer Than D Version* • Development system: The developer does not see the object version for delivery. Either an old object has been saved in the system as a D version or is not saved correctly due to incorrect system settings. • Test system: The tester sees incorrect versions. Either A versions were imported or D versions are activated in the D versions.	SAD	1
	Object Status: D Version Newer Than A Version • Content activation failed or has not yet taken place.	SDA	1
	Object Status: No A Version Available • Content activation failed or has not yet taken place.	SNA	1
	Object Status: No D Version Available *SND is only executed for development systems because there is no D version in test systems.* • Writing the D version does not work. The system settings should be checked.	SND	1

For more information on the various object versions, look in the SAP Library under *SAP NetWeaver → Information Integration → SAP Business Information Warehouse → Data Warehousing → Data Warehouse Management → Business Content (Versions)*.

Type: Checking for Object Collection Errors (Multi)

Type ID	Assigned Check Categories	Categ. ID	Standard Priority
CE	*Object Collection – Collection Error* *CCE should be executed for test systems.* • The object links to an object that does not exist. Not every object can be collected in the data flow, because ○ , due to the assignment to package $t TMP, whose import from the development system failed or ○ the Repository Cache was not updated. In this case, you update the cache using transaction RSOR → *Extras → Repository Cache (DB) → Fill (Objects*	CCE	1

	form Delivery). The object collection or activation using transaction RSOR leads to an error message.		
	Object Collection – Temporarily Saved Objects *COT should be executed for development systems.* • The object references a locally saved (package $TMP) object. As soon as the object is transported into a new system, the local object disappears. • Change the development class of the locally saved object to a transportable development class or ensure that the collected object is not transported.	COT	1

Type: Checking for Inconsistent Naming Conventions(Multi)

Type ID	Assigned Check Categories	Categ. ID	Standard Priority
NC	*Naming Convention Ignored* • The object does not following the naming conventions for partner and customer namespaces. Under *Settings* → *Global Settings*, determine which of the namespaces is to be checked. In addition: Only when you have released packages for the objects can a naming convention check be done in a customer system. For information on releasing packages, see Releasing Packages [Page 786]. In this case, you should select the option *AND Linking* in the *Global Settings*. • Change the object name.	ANC	2

Type: Checking for an Inactive Transfer Structure

Type ID	Assigned Check Categories	Categ. ID	Standard Priority
TA	*Inactive Transfer Structures* • The transfer structure was either changed manually or through the replication of assigned DataSources. As long as reactivation has not occurred (transaction RSA1 → *InfoSources*), loading of data and the objects that are part of the import failed.	ATA	4

Type: Checking for InfoObjects without an InfoObject Catalog

Type ID	Assigned Check Categories	Categ. ID	Standard Priority
IC	*InfoObject without InfoObject Catalog* • When the InfoObject was created, there was no assignment to an InfoObject catalog. If the	AIC	2

	assignment has not been done, the InfoObject does not follow the Content standards and is assigned to one of the following standard categories: ○ 0CHANOTASSIGNED ○ 0KYFNOTASSIGNED ○ 0TIMNOTASSIGNED ○ 0UNINOTASSIGNED		

Type: Checking for Inconsistent Roles

Type ID	Assigned Check Categories	Categ. ID	Standard Priority
RL	*Roles to Which Unavailable Objects Are Assigned* • A object was deleted but not its assignment to a role. This leads to inconsistencies and collection errors as long as the assignment was not also deleted in transaction PFCG. You can use transaction PFCG → *Menu* to check the existing object references to the role. You can check the existence of the referenced object using its name. The transaction RSRT (for query elements) or the report RS_TEMPLATE_MAINTAIN (for templates) can also be helpful here.	ARL	1

Type: Checking for Query Elements with Duplicate GUIDs

Type ID	Assigned Check Categories	Categ. ID	Standard Priority
DG	*Query Elements with Duplicate GUIDs* • A query element is defined by a. its technical name, for example 0SBO_C11_Q003, which you enter manually. b. the COMPUID (GUID), the unique object key, for example 037OLMX9GF8CK9P23IV2MDO1H, which is generated automatically. If queries with identical technical names are generated in different BI systems, these elements have a different GUID. This means that 2 GUIDs exist for the identical technical name during a transport of both objects into a common system. System A: ▪ Technical name: ABC_Q001	ADG	2

	▪ GUID : 123456789 System B : ▪ Technical name: ABC_Q001 ▪ GUID : 987654321 After transport into system C: ▪ 2 objects for the technical name ABC_Q001 ▪ GUID object 1 : 123456789 ▪ GUID object 2 : 987654321 This means that inconsistencies will occur during query executions because the system selects one of the two GUIDs randomly. • Within the system landscape, check table RSZCOMPDIR by entering the technical name of the query element in field *COMPID.* You will see multiple objects displayed for different versions with different GUIDs. Try to determine the correct object version using the Content time stamp. Now proceed according to SAP Note 541024 and change the technical name of one of the objects. • If you are sure which object is incorrect, you can also delete this object using transaction RSZDELETE by selecting a COMPID. Deleted objects cannot be restored. Deletion in a development system also has to be exported so that the problem is eliminated in the follow-on system.		

Type: Checking for Routines That Refer to Fixed, Programmed Structures

Type ID	Assigned Check Categories	Categ. ID	Standard Priority
BI	*Routines with Fixed, Programmed Structures (/BI*/*)* • If there are references to generated BW-DDIC structures or types such as /BIC/* in routines, it can lead to problems during the transport or during Content installation (RSOR). If the referenced structures do not yet exist at the time of import or Content installation because the affected BI object is not active in the system, it could lead to termination. • The routines should be programmed dynamically. These dynamic assignments could	ABI	1

	appear as follows: *.. data definitions for internal table and work area data: lt_data type ref to data. field-symbols: <t_data> type standard table. *.. store name of structure (infocube 0FIGL_C01) as string constants: c_tabname type rstlogotab value '/BI0/V0FIGL_C012'. *.. create internal table and assign it to <t_data> create data lt_data type table of (c_tabname). assign lt_data->* to <t_data>.		

5.1.2 Analysis and Repair Environment

Purpose

In the analysis and repair environment, you can perform consistency checks on the data and metadata stored in a BI system. This predominantly tests the foreign key relationships [External] between the individual tables of the enhanced star schema of the BI system.

Integration

You can call the analysis and repair environment in the following ways:

- On the *SAP Easy Access* screen, choose *Administration* → *Analysis Tool* (transaction code *RSRV*).
- On the *SAP Easy Access* screen, choose *Modeling* → *Object Maintenance* → *InfoObject* (Transaction RSD1). Select the InfoObject you want.
  - Choose *Analyze* from the initial screen.

  Or:

  - Choose *Maintain*. You get to the maintenance screen. Choose *Edit* → *Analyze InfoObject*.

  You get to the *Analysis and Repair of BW Objects* screen.

  If you are using the analysis and repair environment for the first time you will see the message “values have not been found for all setting parameters”. This warns that no settings have yet been saved for your user

Features

The screen *Analysis and Repair of BI Objects* comprises two separate screen areas:

1. In the left-hand screen area you see a hierarchy with the list of available tests.

   You can select elementary or compound tests:

   - An **elementary test** is a test that cannot be subdivided into smaller tests and therefore can only be executed as a whole. Elementary tests relate to master data and transaction data, DataStore objects, hierarchies, a database (indexes, parameters, statistics), aggregates, PSA tables, and documents.
   - A **combined test** determines which elementary tests are to be performed according to the parameters entered.

     You can display a short description of a test by choosing *Description* from the context menu of a test.

2. The right-hand screen area is empty to begin with. The tests you select are recorded here.

   A selection of tests is referred to as a **test package**.

Compiling and Executing a Test Package

1. Select the test you want. You can transfer an individual test or a whole folder with all the tests that exist in the hierarchical structure of this folder in your test package in the following ways:

   - By double-clicking on the required test or folder in the left-hand side of the screen
   - With Drag&Drop, by transferring the required test or folder from the test list to the right-hand screen area

   Each selected test is displayed in the right-hand side of the screen as a closed folder.

   An exception to this is elementary tests without parameters. These do not appear as folders.

2. Provide parameters for the tests. You can start parameter input in the following ways:

   - By double-clicking a test (a test package)
   - By expanding the folder for the test

     You do not need to provide parameters for tests that do not have them. Take note of the appropriate message after selecting the test.

   The *Parameter Input* dialog box appears. Enter the required parameters. Input help is available in many cases.

   After parameters have been entered, a folder with the name *Parameter* is inserted under the test in question. This contains the values of the parameter. In some circumstances the name of the test is also changed so that you can see, at a glance, the parameter values for which the test is to be performed. This is particularly helpful if you have selected the same test several times and want to provide different parameters for it.

If you have provided parameters for a combined test an additional *Elementary Tests* folder is inserted along with the *Parameter* folder. This folder contains the elementary tests from which the combined test is built.

You can delete individual elementary tests from the test package by using Drag&Drop to move this back to the list of tests. You can reduce the runtime of the test in this way.

3. After you have provided parameters for all tests, you can start the test run. Choose *Execute*. During execution the symbol for the test is changed from :
   - To if no errors occurred for the test
   - To if warnings only were output
   - To if the test produced errors

   To delete all elementary tests from the current test package that did not output errors or warnings, choose *Deselect Tests Without Errors*. If you want to delete the whole test package currently selected, choose *Delete*.

Test Output

The output of the test is written to the application log and stored in the database. This means they are available for comparison with subsequent test runs.

Settings can also be made for the system so that the log is automatically displayed after a test run (see under “Settings”). Otherwise, choose the *Display* function from the *Analysis and Repair of BI Objects* screen. The *Log* screen appears.

The left-hand area of the screen contains an overview of the most recently performed test runs. By double-clicking on a folder, the system displays all messages hanging under this node as a flat list in the right-hand screen area.

As long as it is available, you can display further information using the following functions:

Detail Information

Long Text for Message

Technical Information

Repairs

Some tests can repair any errors and inconsistencies that are discovered. However, automatic corrections are not generally possible.

If, for example, entries are missing in the SID table for a characteristic but the lost SIDs are still being used in a dimension table of an InfoCube (and the corresponding dimension key in the fact table), this can only be corrected by loading the transaction data for the InfoCube again.

You must also remember that the execution of repairs depends on you executing the repairs in the right order.

We recommend that you always read the test documentation first and obtain a clear idea of where the errors lie, before you execute repairs.

To perform repairs, return to the *Analysis and Repair of BI Objects* screen. If you want to start a repair, choose *Remove Error*. As the dataset may have changed between running the analysis and the repair, the necessary tests are executed again before the actual repair. The test output can be found in the application log.

Additional Functions on the *Analysis and Repair of BI Objects* Screen

Test Package Menu

When you leave the analysis and repair environment, you lose the test package you see in the display. However, you can save this: Choose *Edit* → *Test Package* → *Save* (see under “Menu Application Log“).

You can select further functions under *Edit* → *Test Package*:

- *Load* (load test package)
- *Edit* (load test package for editing; the test package is then locked against external changes)
- *Delete*
- *Schedule* (schedule test package to be executed subsequently or regularly in background processing; see under "Jobs Menu")

Jobs menu

If you have scheduled a test package and want to monitor its status, choose *Jobs* → *Job Overview*.

Application log menu

This option allows you to display old logs, both from previous test runs and from scheduled test runs. You also have the option to delete test logs here.

Settings menu

You can make user-specific settings and save these settings using *Settings* → *Change Settings*.

Setting	Selection
Display Result After Test Run	Always / With Errors / Never
Screen Size	Adjust the screen areas: Control the size of the right and left-hand areas
Application Log	Modify the layout, specify an expiry date and a delete condition

The settings are read automatically when you start the test environment.

If no values are available for the settings options, the user will be notified with a message.

Executing Test Packages in Process Chains

You insert ABAP program RSRV_JOB_RUNNER to your process chain in process chain maintenance (transaction code RSPC). (To do this, select the process type “ABAP Program“ with Drag&Drop under “General Services” in the process type view.) To edit process variants, you have to specify the program name and the name of a program variant. Enter RSRV_JOB_RUNNER as the program name and choose a program variant name. Choose *Change*. On the following screen you can either change or display variants that already exist, or create a new variant. When creating a new variant you have to specify the following:

- The package name (input help is available).
- The level of detail in which the log is to be integrated into RSRV in the process chain log.
- The message type that signifies that the processing of the process chain is to be terminated.

The log for the RSRV process in the process chain consists of the following parts:

- A summary that specifies whether errors, warnings, or no errors at all where output for each individual elementary test.
- A view of the log for the test package, in the level of detail specified.

  If you have chosen the value 3 for the level of detail, only messages up to level 3 are included in the log of process chain processes. Messages that have been output at a lower level of a test in the test package are not displayed in this log.

  Note that (unlike the application log) in the processes log errors are **not** propagated from levels of detail to lower levels of detail. If, for example, a single error was output on level 4, it would be recorded in the summary that the corresponding test had delivered an error, but this error would not be recorded in the second part of the log.

A complete log is always written, independently of the RSRV processes log in the process chain. You can display this with *Application Log → Display Log → From Background Processing*.

Constraints

Note that there is no transport object for test packages. Therefore they cannot be transported. Process chains that execute RSRV test packages therefore have to be reworked manually after a transport to another system. The corresponding test packages have to be created.

5.2 Debugging

Purpose

Update Simulation in the Monitor

You can simulate posting before you post data into InfoProviders from the PSA or ALE inbound processing, or if the update to the InfoProviders failed during the load process. The debug option for the transformation rules supports you in error finding.

For more information, see Update Simulation in the Monitor [Page 797]

Debugging in the Data Transfer Process

You can execute a standard data transfer process in debug mode.

For more information, see Creating Data Transfer Processes [Page 322].

5.2.1 Update Simulation in the Extraction Monitor

Use

Before you post the data from the PSA or ALE inbound processing into the InfoProvider, or if the update into the InfoProvider failed during the loading process, you can simulate the posting. The debug option for the transfer rules and update rules supports you when searching for errors in case the update into the InfoProvider is terminated in the transfer or update rules.

If you have switched on the error handling using the scheduler (*Update Parameter* tab page), an error is logged in the monitor if the field content in a data record cannot be transferred from the transfer structure format to the communication structure format. The error message provides detailed information about which field value in the data record cannot be converted. We recommend that you switch on the consistency check for characteristic values in the scheduler (*Processing* tab page) in addition to error handling

If the data record correctly exists both in both the source system and the PSA, you can execute the error analysis for the transfer rules by simulating an update for a data package in the monitor. In the breakpoint that was set as the standard setting, you find additional information that supports you when searching for errors with the debug function for the transfer rules.

If the error appears first in the update rules, the system likewise logs an error message. You then have the option of debugging the update rules for the error analysis.

Prerequisites

The data was successfully received in BI.

Features

You can simulate the update on a data package level. In doing so, you can define for which data target the simulation needs to be executed and which records need to be used for the simulation.

Activities

1. Select the data request that you want to examine in the monitor.
2. Select *Simulate Update* on the *Detail* tab page for a data package by using the context menu (right click).

   The *Simulation Selections* dialog box appears.
3. Select the data target for which the simulation needs to be executed.
4. If you want to switch the debugging on in the transfer or update rules, select the appropriate field.
5. Specify whether you want to use all the data package records for the simulation or if you only want to select specific records.
6. Choose *Execute Simulation*.

7. Select the records to be examined from the following dialog box where necessary and confirm your entries.

   You see the data in the communication structure format. In the case of data with flexible updating, you can change to the view for data target data records. In the data target screen you can display the records belonging to the communication structure for selected records in a second window.

   If you have activated debugging, the *ABAP Debugger* appears and you can execute the error analysis there.

6 Reference

This section contains a list of all available APIs and tutorials, along with check lists that help you to avoid design errors.

6.1 Interface Overview

The following overview of the various interfaces in BI shows the most important properties of the interfaces and can therefore help you with decision-making.

Interface	**Service:** **R**ead **W**rite **D**isplay *	**Programming Language**	**Mass Data Support**	**B**ackground**/** **D**ialog	**Type**	**Prerequ**
DB Connect	W	ABAP	X	D/B	Tool	DataSou
UD Connect	W	Java	-	D/B	Programming	DataSou
BI Service API	W	ABAP	X	D/B	Programming	DataSou
File Interface	W	ABAP	X	D/B	Tool	DataSou
Web Service for Staging	W	ABAP/Java	X	B	API	InfoProv
XI Integration	R/W	ABAP	X	B	Tool	InfoProv
Staging BAPIs	W	ABAP	X	D/B	Programming	InfoProv
DataStore Object APIs	W	ABAP	X	D	API	DataSto
Interface for Real-Time InfoCubes	W	ABAP	X	D	API	InfoCube
APIs for Master Data	R/W	ABAP	-	B	API	Master d
APIs for Hierarchies	R/W	ABAP	-	B	Programming	Master d
VirtualProviders with Function Modules	R	ABAP	-	D	Programming	InfoProv
Open Hub Destination	R	-	X	B	Tool	InfoProv
Open Hub Destination	R	ABAP	X	B	Programming	InfoProv

APIs						
Data Mart Interface	R	ABAP	X	D	API	InfoProv
OLE DB for OLAP	R	Various	-	D	Programming	Query/In
MDX OLAP BAPIs	R	ABAP	-	D	Programming	Query/In
XML for Analysis	R	Java	-	D	Programming/API	Query/In
Web Service for Accessing Query Data	R	ABAP	-	D	Programming	Query
Web Design API	D	-	-	D	Tool	Web tem
Visual Composer	D	Java	-	D	Tool	Query/V
Content from BI as iView in Portal	D	-	-	D	Tool	Query/W
Content from BI as Callable Object in Guided Procedures	D	-	-	D	Tool	Query/W
BI Java SDK	R	Java	-	D	Programming/API	Query/In
Analysis Process Designer	R	ABAP	X	B	Tool	Query/In Master d

* READ returns data that can be processed further; DISPLAY renders data but you cannot edit it.

These interfaces are included in scenarios *Enterprise Data Warehousing*, *Enterprise Reporting, Query and Analysis,* and *Business Planning and Analytical Services*.

For a detailed description, see the Developer's Guide for the relevant scenario under *Reference*:

Enterprise Data Warehousing [Page **Error! Bookmark not defined.**]

Enterprise Reporting, Query, and Analysis [External]

Business Planning and Analytical Services [External]

If you extract data from your BI system into non-SAP systems, you must have this scenario certified – irrespective of the tool you are using for the extraction, whether it be an open hub destination, list output (LISTCUBE), BI APIs, Analysis Process Designer, or download to Microsoft Excel.

6.1.1 DB Connect

You use DB Connect to open other database connections in addition to the default connection and use these connections to transfer data from tables or views into a BI system.

Interface Properties:

- Write interface
- Programming language is ABAP

- Mass data is not supported
- Execution in dialog/background
- Programming proficiency is not required
- A DataSource/InfoProvider is a prerequisite

For more information, see Data Transfer with DB Connect [Page 603].

6.1.2 UD Connect

UD Connect (Universal Data Connect) uses the Application Server J2EE connectivity to enable the reporting and analysis of both SAP and non-SAP data. Using UD Connect, you can access all relational and multidimensional data sources. UD Connect transfers the data as flat data.

Interface Properties:

- Write interface
- Programming language is Java
- Mass data is not supported
- Execution in dialog/background
- Programming proficiency is not required
- A DataSource/InfoProvider is a prerequisite

For more information, see Transferring Data with UD Connect [Page 576].

6.1.3 BI Service API

Using the Service API, you can send data by pull to the BI system.

For connecting to SAP systems, there are defined BI Content interfaces. These are used by application-specific extractors, each of which is hard-coded for the DataSource that was delivered with BI Content.

API	Use
Generic extractor	Customer-defined extractors can be developed

Interface Properties:

- Write interface
- Programming language is ABAP
- Mass data is supported
- Execution in dialog/background
- Programming proficiency is not required (except for generic extractor with data retrieval API).
- A DataSource (except for Web Service DataSource) or an InfoProvider is a prerequisite.

More Information:

You can refer to an example in the system in the ABAP Editor: RSAX_BIW_GET_DATA_SIMPLE.

We recommend using BI Content DataSources.

For more information, see

Documentation about API for the generic extractor: Function Module: Interface Description and Procedure [Page 546]

Documentation about Service API (Data Extraction): Data Extraction from SAP Source Systems [Page 523]

6.1.4 File Interface

Using this interface, you can access files in ASCII format or CSV format.

Interface Properties:

- Write interface
- Programming language is ABAP
- Mass data is not supported
- Execution in dialog/background
- Programming proficiency is not required
- A DataSource/InfoProvider is a prerequisite

For more information, see Transferring Data from Flat Files [Page 622].

6.1.5 Web Service for Staging

You use the Web service to write the data from the source into the PSA. The transfer of data is controlled externally, without placing demand on BI.

You can use this interface in conjunction with real-time data acquisition.

Interface Properties:

- Write interface
- Programming language is ABAP
- Mass data is supported
- Execution in dialog (data is passed in conjunction with RDA for reporting and analysis).
- Programming proficiency is not required for generating the interface.
- A Web service DataSource is a prerequisite

For more information, see

Data Transfer Using Web Services [Page 571]

Transferring Data Using Web Services [Page 638]

6.1.6 XI Integration

Based on inbound and outbound messages, application data from applications can be distributed to the BI system using XI.

The XI inbound message gets application data using the RFC, IDoc, or http (XML) protocol. Using mapping programs, data is bundled into a XI outbound message on the integration server. This message generates an ABAP proxy on the BI server. In this method, data can be written to the BI system.

Interface Properties:

- Write interface
- Programming language is ABAP
- Mass data is supported
- Execution in background
- Programming proficiency is required
- Data can be updated into all types of InfoProviders

6.1.7 Staging BAPIs

BI provides open interfaces, or staging BAPIs, to enable the extraction of data and metadata from non-SAP sources at application level.

Interface Properties:

- Write interface
- Programming language is ABAP
- Mass data is not supported
- Execution in dialog/background
- Programming proficiency is not required
- A DataSource/InfoProvider is a prerequisite

The staging BAPIs are currently supported for 3.x DataSources.

For more information, see Transferring Data from Non-SAP Systems [Page 628].

6.1.8 DataStore Object APIs

The following APIs are available for DataStore objects for direct update. They allow you to fill DataStore objects with data from external applications:

API	Use
BAPI_ODSO_READ_DATA_UC	Reads data from all DataStore objects
RSDRI_ODSO_INSERT	Inserts new data (with keys not yet in the system)
RSDRI_ODSO_INSERT_RFC	Inserts new data (with keys not yet in the system). It can be called remotely.

RSDRI_ODSO_MODIFY	Inserts data with new keys. For data that already has keys, the data is changed.
RSDRI_ODSO_MODIFY_RFC	Inserts data with new keys. For data that already has keys, the data is changed. It can be called remotely.
RSDRI_ODSO_UPDATE	Changes data that already has keys in the system
RSDRI_ODSO_UPDATE_RFC	Changes data that already has keys in the system. It can be called remotely.
RSDRI_ODSO_DELETE_RFC	Deletes data

Interface Properties:

- Write and read interface
- Programming language is ABAP
- Mass data is supported
- Execution in dialog
- Programming proficiency is required
- A DataStore object for direct update is a prerequisite (except for the read API BAPI_ODSO_READ_DATA_UC)

More Information:

For more information, see DataStore Data and External Applications [Page 163].

The APIs are described in the system.

6.1.9 Interface for Real-Time InfoCubes

You use this RFC-enabled interface to write transactional data for each request directly to an InfoCube.

API	Use
RSDRI_CUBE_WRITE_PACKAGE_RFC	Writes data for each request to an InfoCube

Interface Properties:

- Write interface
- Programming language is ABAP
- Mass data is supported
- Execution in background (no high frequency, for example, per minute)
- Programming proficiency is required
- Data is updated to an InfoCube

More Information:

Note that this BAPI is not released.

Furthermore, note that this BAPI cannot be used for frequent calls. Each new call potentially generates a new request. This can lead to problems in request administration. In particular this may lead to problems with performance, for example while reading request information.

6.1.10 APIs for Master Data

For master data InfoObjects, APIs are available that enable you to write, read, change, and delete data in the master data.

API	Use
RSNDI_MD_ATTRIBUTES_UPDATE	Writes attributes using a fixed transfer structure
RSDMD_WRITE_ATTRIBUTES_TEXTS	Writes attributes and texts using a generic transfer structure
RSNDI_MD_TEXTS_UPDATE	Writes texts using a fixed transfer structure

Interface Properties:

- Write and read interface
- Programming language is ABAP
- Mass data is not supported
- Execution in dialog
- Programming proficiency is required
- An InfoObject is a prerequisite

More Information:

The APIs are described in the function module documentation.

There are BAPIs for the metadata of master data. These are described in the BAPI transaction in the system.

6.1.11 APIs for Hierarchies

For InfoObjects, APIs are available that enable you to write, read, change, and delete hierarchies.

API	Use
RSNDI_SHIE_ACTIVATE	Activates hierarchy
RSNDI_SHIE_CATALOG_GET	Lists all hierarchies
RSNDI_SHIE_DELETE	Deletes hierarchy
RSNDI_SHIE_MAINTAIN	Maintains hierarchy
RSNDI_SHIE_STRUCTURE_GET3	Reads hierarchy
RSNDI_SHIE_STRUCTURE_UPDATE3	Writes hierarchy
RSNDI_SHIE_SUBTREE_DELETE	Deletes subtree

Interface Properties:

- Write and read interface
- Programming language is ABAP
- Mass data is not supported
- Execution in dialog
- Programming proficiency is required
- An InfoObject with hierarchies is a prerequisite

More Information:

The APIs are described in the function module documentation in the system.

6.1.12 VirtualProviders with Function Modules

InfoProviders with transaction data that is not stored in the object itself, but which is read directly for analysis and reporting purposes. The relevant data can be from the BI system, or other SAP or non-SAP systems.

VirtualProviders only allow read access to data.

Interface Properties:

- Remote/real-time interface
- Programming language is ABAP
- Mass data is not supported
- Execution in dialog
- Programming proficiency is not required (except for the function module interface)
- A suitable DataSource for direct access is a prerequisite (except for the function module interface)

More Information:

For more information, see VirtualProviders with Function Modules [Page 171].

See also the How-To Guide in the SAP Developer Network: *How To... Implement a VirtualProvider with Services*.

In the system documentation, various types of APIs are described.

6.1.13 Open Hub Destination

The open hub destination allows you to distribute data from a BI system to non-SAP data marts, analytical applications, and other applications. It allows you to ensure controlled distribution across several systems.

Interface Properties:

- Read interface
- Programming language is ABAP

- Mass data is supported
- Execution in background
- Programming proficiency is not required
- Data has to be in an InfoProvider

For more information, see Open Hub Destination [Page 651].

See also Open Hub Destination APIs [Page 806].

6.1.14 Open Hub Destination APIs

You can use the open hub service to extract data to non-SAP systems. Various APIs allow you to connect a third-party tool (such as Ascential) to the BI system and then distribute the data to other non-SAP systems.

API	Use
RSB_API_OHS_DEST_SETPARAMS	You use this API to transfer the parameters of the third-party tool that are required to extract data to the BI system. These parameters are saved in a parameter table within the BI system in the metadata for the open hub destination.
RSB_API_OHS_3RDPARTY_NOTIFY	This API sends a message to the third-party tool after extraction. It transfers the open hub destination, the request ID, the name of the database table, the number of extracted data records and the time stamp. In addition, you can add another parameter table that contains the parameters that are only relevant for the third-party tool.
RSB_API_OHS_REQUEST_SETSTATUS	This API sets the status of the extraction to the third-party tool in the open hub monitor. Red means that the existing table is not overwritten. If the status is green, the request is processed further.
RSB_API_OHS_DEST_GETLIST	This API delivers a list of all open hub destinations.
RSB_API_OHS_DEST_GETDETAIL	This API identifies the details of an open hub destination.
RSB_API_OHS_DEST_READ_DATA	This API reads data from the database table in the BI system.

Interface Properties:

- BI read interface
- Programming language is ABAP
- Mass data is supported
- Execution in background
- Programming proficiency is required
- Data has to be in an InfoProvider

For more information, see Third-Party Tools As Destinations [Page 654].

6.1.15 Data Mart Interface

The RSDRI_INFOPROV_READ function module reads data from the InfoProvider. The technical name of the InfoProvider is specified in the I_INFOPROV parameter. Supported InfoProviders are currently InfoCubes, DataStore objects, and MultiProviders.

API	Use
RSDRI_INFOPROV_READ_RFC	Reads data from an InfoProvider

Interface Properties:

- Read interface
- Programming language is ABAP
- Mass data is supported (check that the interface is used efficiently: parallel calls, filter criteria, buffering, and so on.)
- Execution in dialog
- Programming proficiency is required
- Data has to be in an InfoProvider

More Information:

The APIs are described in the function module documentation.

The ABAP report, RSDRI_INFOPROV_READ_DEMO, contains an example of how a function module can be used. You can find this in the ABAP Editor.

6.2 Checklist for Developing BI Objects

What you need to check…

Developing the data model:

	Is the data model clear, integrated and consistent?
	Have you structured your system in a data warehouse layer architecture? Have you adhered to the recommendations for data modeling?

Developing InfoObjects:

	Are you planning to develop new characteristics? Characteristics should be reusable, where possible. If you just need a new description for an existing characteristic, create the new characteristic as a reference to the existing characteristic.
	Do you need a special key figure to count the occurrences of a characteristic?

Using time characteristics:

	If you are using 0CALDAY in an InfoProvider, you should also include the higher-level time characteristics in the InfoProvider (for example, 0CALWEEK, 0CALMONTH, 0CALQUARTER, 0CALYEAR).

	If you are using 0FISCVARNT and 0CALDAY in an InfoProvider, you should also include 0FISCPER and 0FISCYEAR.

Developing InfoSources:

	Do the InfoSources only contain InfoObjects that are technically relevant or relevant for analysis and reporting?

Developing InfoProviders:

	Are you using DataStore objects? You should use DataStore objects if you need the history or analysis of an individual record. InfoCubes, on the other hand, can be aggregated and therefore have better performance.
	Are you using InfoSets with time-dependent master data? In this case, you should not define queries with time intervals.
	Are you using MultiProviders? Make sure that all the characteristics you want to use in the query are contained in all the individual InfoProviders.

Defining the load process:

	Have you defined transformation rules so that master data can be read? With reverse postings for delta-enabled DataSources, note that master data should not be read in the transformation rule for an InfoCube. The system can only process reverse postings upon reading master data correctly in the transformation rules for a DataStore object.
	Are you using process chains? You should use process chains to perform each load and further processing process.