How to…

Report on cross company stock in transit

BUSINESS INFORMATION WAREHOUSE

Applicable Releases: BW >= 2.0B / R/3 >= 4.0B

SAP (SAP America, Inc. and SAP AG) assumes no responsibility for errors or omissions in these materials.

These materials are provided “as is” without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall not be liable for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials.

SAP does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within these materials. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third party web pages nor provide any warranty whatsoever relating to third party web pages.

mySAP BI “How-To” papers are intended to simplify the product implementation. While specific product features and procedures typically are explained in a practical business context, it is not implied that those features and procedures are the only approach in solving a specific business problem using mySAP BI. Should you wish to receive additional information, clarification or support, please refer to SAP Professional Services (Consulting/Remote Consulting).

May 2004

1 Business Scenario

Current stock reporting in BW (e. g. InfoCube 0IC_C03) doesn't offer the possibility to report on stock in transit that is created via purchase orders/sales orders (Cross company stock in transit - movement types 101 and 643 for cross company).

2 Introduction

Current extractors like 2LIS_03_BF cannot report on this as the material movements come with "normal movement" types (101) that are also used by stock transfers with company external customers/vendors.

Such information could only be deliverd by 2LIS_03_BF due to the posting logic.

If the purchase orders history (table EKBE) is additionally read in the extraction process (purchase orders with item category "0") then this information could be obtained. But his solution is not feasible as this would lead into big performance problems.

Therefore we prefer a solution that extracts cross company stock directly into BW.

This extrator transfers information displayed with R/3 transaction MB5T (Display of stock in transit) into BW.

This data can be loaded as a daily snapshot into an ODS (for less data amount and preferred reporting on single records) or InfoCube (for many data and preferred reporting on aggregated data) and then be combined with 0IC_C03 in a MultiProvider scenario for consolidated stock reporting.[1]

[1] The following example uses an ODS object.

3 The Step By Step Solution

Overview of the steps:

R/3: Creation of DDIC structures, etc.

Creation of function module extractor, etc.

Creation of DataSource

DataSource testing and comparison with MB5T

BW: Creation of missing InfoObjects

Creation of InfoSource, ODS and update rules

Creation of a sample query

Date Source replication; creation of transfer rules and data extraction from R/3

Query testing and comparison with MB5T

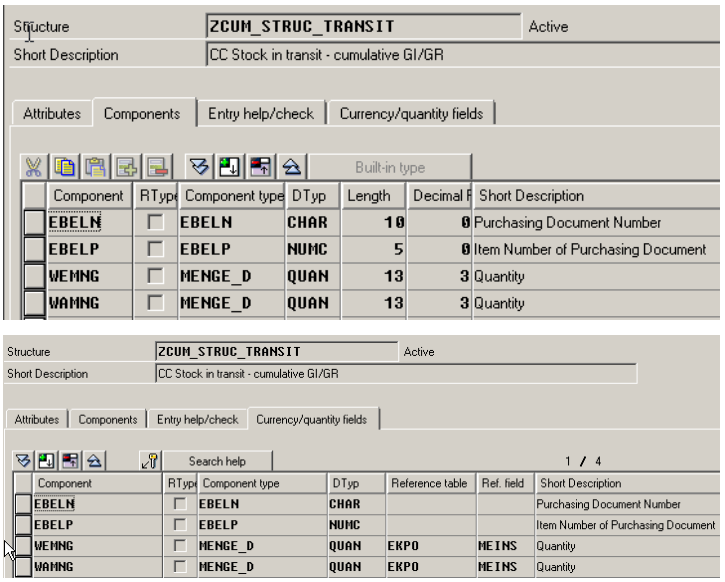
Creation of a Multi Provider and Multi Provider Query

Query testing of consolidated stock reporting

1. Steps in R/3

1. Creation of DDIC structures, etc.

1. ZCUM_STRUC_TRANSIT



2. ZSTOCK_TRANSIT

Structure: ZSTOCK_TRANSIT — Active

Short Description: CC Stock in transit

Attributes | Components | Entry help/check | Currency/quantity fields

Built-in type — 1 / 22

Component	RType	Component type	DTyp	Length	Decimal F	Short Description
MATNR		MATNR	CHAR	18	0	Material Number
WERKS		WERKS_D	CHAR	4	0	Plant
BSTAUS		BSTAUS	CHAR	1	0	Stock values in the Logistics Information System
BSTTYP		BSTTYP	CHAR	1	0	Stock categories in the Logistics Information System
MENGE		MENGE_D	QUAN	13	3	Quantity
MEINS		MEINS	UNIT	3	0	Base Unit of Measure
DMBTR		DMBTR	CURR	13	2	Amount in local currency
HWAER		WAERS	CUKY	5	0	Currency Key
PSTYP		PSTYP	CHAR	1	0	Item category in purchasing document
SATNR		SATNR	CHAR	18	0	Cross-Plant Configurable Material
ATTYP		ATTYP	CHAR	2	0	Material category
VKORG		VKORG	CHAR	4	0	Sales Organization
VTWEG		VTWEG	CHAR	2	0	Distribution Channel
PRICE_DATE		DATS	DATS	8	0	Field of type DATS
BUKRS		BUKRS	CHAR	4	0	Company Code
VLFKZ		VLFKZ	CHAR	1	0	Plant category
MTART		MTART	CHAR	4	0	Material Type
BUDAT		BUDAT	DATS	8	0	Posting Date in the Document
EBELN		EBELN	CHAR	10	0	Purchasing Document Number
EBELP		EBELP	NUMC	5	0	Item Number of Purchasing Document
ETENR		ETENR	NUMC	4	0	Schedule line
RESWK		RESWK	CHAR	4	0	Supplying (issuing) plant in case of stock transport order

Attributes | Components | Entry help/check | Currency/quantity fields

Search help — 1 / 22

Component	RType	Component type	DTyp	Reference table	Ref. field	S
MATNR		MATNR	CHAR			Ma
WERKS		WERKS_D	CHAR			Pla
BSTAUS		BSTAUS	CHAR			St
BSTTYP		BSTTYP	CHAR			St
MENGE		MENGE_D	QUAN	ZSTOCK_TRANSIT	MEINS	Qu
MEINS		MEINS	UNIT			Ba
DMBTR		DMBTR	CURR	ZSTOCK_TRANSIT	HWAER	An

3. ZSTOCK_TRANSIT_SEL

Structure: ZSTOCK_TRANSIT_SEL — Active
Short Description: CC Stock in transit - selections

Attributes | Components | Entry help/check | Currency/quantity fields

Built-in type — 1 / 32

Component	RType	Component type	DTyp	Length	Decimal P	Short Description
UMREZ		UMREZ	DEC	5	0	Numerator for Conversion to Base Units of Measure
UMREN		UMREN	DEC	5	0	Denominator for Conversion to Base Units of Measure
WAMNG		WAMNG	QUAN	13	3	Issued quantity
WEMNG		WEMNG	QUAN	13	3	Quantity of goods received
RETPO		RETPO	CHAR	1	0	Returns item
NETWR		NETWR	CURR	15	2	Net value in document currency
WAERS		WAERS	CUKY	5	0	Currency Key
BSTMG		BSTMG	QUAN	13	3	Purchase order quantity
WKURS		WKURS	DEC	9	5	Exchange rate
.INCLUDE		ZSTOCK_TRANSIT		0	0	CC Stock in transit

Structure: ZSTOCK_TRANSIT_SEL — Active
Short Description: CC Stock in transit - selections

Attributes | Components | Entry help/check | Currency/quantity fields

Search help — 1 / 32

Component	RType	Component type	DTyp	Reference table	Ref. field
UMREZ		UMREZ	DEC		
UMREN		UMREN	DEC		
WAMNG		WAMNG	QUAN	ZSTOCK_TRANSIT_SEL	MEINS
WEMNG		WEMNG	QUAN	ZSTOCK_TRANSIT_SEL	MEINS
RETPO		RETPO	CHAR		
NETWR		NETWR	CURR	ZSTOCK_TRANSIT_SEL	WAERS
WAERS		WAERS	CUKY		
BSTMG		BSTMG	QUAN	ZSTOCK_TRANSIT_SEL	MEINS
WKURS		WKURS	DEC		
.INCLUDE		ZSTOCK_TRANSIT			
MATNR		MATNR	CHAR		
WERKS		WERKS_D	CHAR		
BSTAUS		BSTAUS	CHAR		
BSTTYP		BSTTYP	CHAR		
MENGE		MENGE_D	QUAN	ZSTOCK_TRANSIT_SEL	MEINS
MEINS		MEINS	UNIT		
DMBTR		DMBTR	CURR	ZSTOCK_TRANSIT_SEL	HWAER

4. Type-Pool ZCON

```
TYPE-POOL zcon.

* Field names
CONSTANTS: zcon_gc_fn_material    TYPE rsfieldnm VALUE 'MATNR',
           zcon_gc_fn_pstyp       TYPE rsfieldnm VALUE 'PSTYP',
           zcon_gc_fn_budat       TYPE rsfieldnm VALUE 'BUDAT',
           zcon_gc_fn_plant       TYPE rsfieldnm VALUE 'WERKS'.

* Ranges constants
CONSTANTS: zcon_gc_sign_include   TYPE char1 VALUE 'I',

           zcon_gc_option_equal   TYPE char2 VALUE 'EQ'.
```

2. Creation of function module extractor, etc.

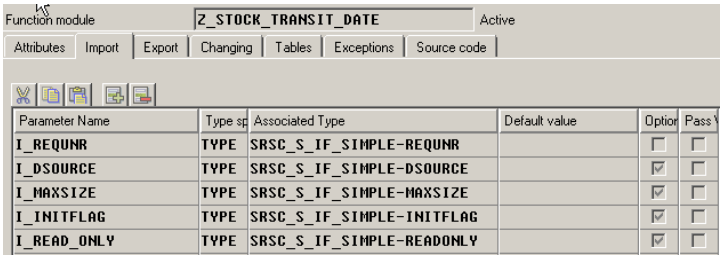
1. Function group ZZ_STOCK_TRANSIT

2. Funtion module Z_STOCK_TRANSIT_DATE

a) Global data

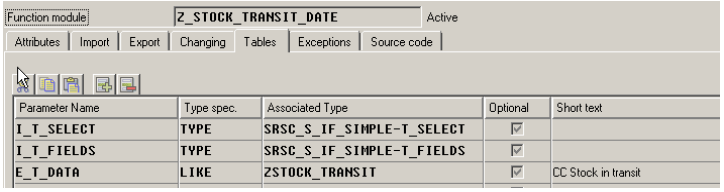
```
FUNCTION-POOL ZZ_STOCK_TRANSIT.             "MESSAGE-ID ..

TYPE-POOLS: SBIWA,
            RSAP,
            SRSC,
            RSAOT,
            RSAZT,
            ZCON.

types: begin of po_num,
         ebeln type ebeln,
         bsart type bsart,
         INCO1 type inco1,
         INCO2 type inco2,
         wkurs type wkurs,
         bedat type bedat,
         ekgrp type ekgrp,
         ekorg type ekorg,
         lifnr type lifnr,
         lifre type lifre,
         waers type waers,
       end of po_num.

types: po_num_t type standard table of po_num.

types: begin of numki_range,
         NRNR   type NRNR,
         range  type ref to data,
       end of numki_range.


data: gt_select    type SRSC_T_SELECT,
      gt_fields    type SRSC_T_FIELDS,
      gt_po_fields   type SRSC_T_FIELDS,

      gt_sel_fields  type SRSC_T_FIELDS.
```

b) Import parameters

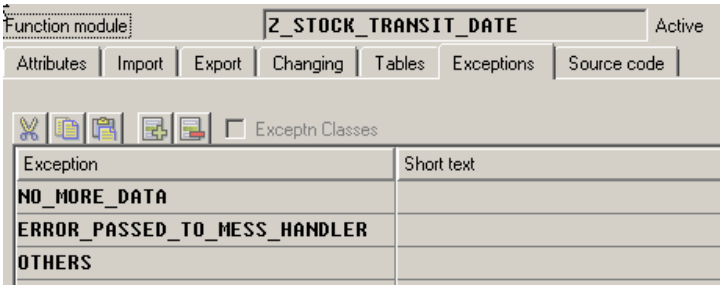


Parameter Name	Type sp	Associated Type	Default value	Option	Pass V
I_REQUNR	TYPE	SRSC_S_IF_SIMPLE-REQUNR		☐	☐
I_DSOURCE	TYPE	SRSC_S_IF_SIMPLE-DSOURCE		☑	☐
I_MAXSIZE	TYPE	SRSC_S_IF_SIMPLE-MAXSIZE		☑	☐
I_INITFLAG	TYPE	SRSC_S_IF_SIMPLE-INITFLAG		☑	☐
I_READ_ONLY	TYPE	SRSC_S_IF_SIMPLE-READONLY		☑	☐

c) Tables parameters



d) Exceptions



e) Source code

```
FUNCTION z_stock_transit_date.
*"----------------------------------------------------------------------
-
*"*"Local interface:
*"  IMPORTING
*"     REFERENCE(I_REQUNR) TYPE  SRSC_S_IF_SIMPLE-REQUNR
*"     REFERENCE(I_DSOURCE) TYPE  SRSC_S_IF_SIMPLE-DSOURCE OPTIONAL
*"     REFERENCE(I_MAXSIZE) TYPE  SRSC_S_IF_SIMPLE-MAXSIZE OPTIONAL
*"     REFERENCE(I_INITFLAG) TYPE  SRSC_S_IF_SIMPLE-INITFLAG OPTIONAL
*"     REFERENCE(I_READ_ONLY) TYPE  SRSC_S_IF_SIMPLE-READONLY OPTIONAL
*"  TABLES
*"      I_T_SELECT TYPE  SRSC_S_IF_SIMPLE-T_SELECT OPTIONAL
*"      I_T_FIELDS TYPE  SRSC_S_IF_SIMPLE-T_FIELDS OPTIONAL
*"      E_T_DATA STRUCTURE  ZSTOCK_TRANSIT OPTIONAL
*"  EXCEPTIONS
*"      NO_MORE_DATA
*"      ERROR_PASSED_TO_MESS_HANDLER
*"      OTHERS
*"----------------------------------------------------------------------
-

  DATA: lr_material  TYPE RANGE OF matnr,
        lr_plant     TYPE RANGE OF werks_d,
        lr_pstyp     TYPE RANGE OF pstyp,
        lr_budat     TYPE RANGE OF budat.

  IF NOT i_initflag IS INITIAL.
    gt_select = i_t_select[].
    gt_fields = i_t_fields[].

    PERFORM budat_validate CHANGING gt_select.

  ELSE.

* transfer selection parameters in ranges

    PERFORM range_fill USING     zcon_gc_fn_material
                                 gt_select
```

```
                        CHANGING  lr_material.

    PERFORM range_fill USING     zcon_gc_fn_plant
                                 gt_select
                        CHANGING lr_plant.

    PERFORM range_fill USING     zcon_gc_fn_pstyp
                                 gt_select
                        CHANGING lr_pstyp.

    PERFORM range_fill USING     zcon_gc_fn_budat
                                 gt_select
                        CHANGING lr_budat.

    PERFORM data_selection_transit_date USING    i_maxsize
                                                 lr_material
                                                 lr_plant
                                                 lr_pstyp
                                                 lr_budat
                                                 gt_sel_fields
                                       CHANGING  e_t_data[].

  ENDIF.

ENDFUNCTION.
```

f) Include LZZ_STOCK_TRANSITF0B

```
*&---------------------------------------------------------------------
*
*&  Include           LZZ_STOCK_TRANSITF0B
*
*&---------------------------------------------------------------------
*

FORM build_range USING    iv_sign
                          iv_option
                          iv_low
                          iv_high
                 CHANGING cr_range    TYPE STANDARD TABLE.

  DATA: lp_data TYPE REF TO data.

  FIELD-SYMBOLS: <ls_fields>    TYPE ANY,
                 <ls_range>     TYPE ANY,
                 <lv_sign>      TYPE ANY,
                 <lv_option>    TYPE ANY,
                 <lv_low>       TYPE ANY,
                 <lv_high>      TYPE ANY,
                 <lv_fieldname> TYPE ANY.

  CREATE DATA lp_data LIKE LINE OF cr_range.
  ASSIGN lp_data->* TO <ls_range>.
  ASSIGN COMPONENT 'SIGN'   OF STRUCTURE <ls_range> TO <lv_sign>.
  ASSIGN COMPONENT 'OPTION' OF STRUCTURE <ls_range> TO <lv_option>.
  ASSIGN COMPONENT 'LOW'    OF STRUCTURE <ls_range> TO <lv_low>.
  ASSIGN COMPONENT 'HIGH'   OF STRUCTURE <ls_range> TO <lv_high>.

  <lv_sign>       = iv_sign.
  <lv_option>     = iv_option.
  <lv_low>        = iv_low.
```

```
  <lv_high>       = iv_high.
  INSERT <ls_range> INTO TABLE cr_range.

ENDFORM.                    " build_range


FORM budat_validate CHANGING   ct_select  TYPE srsc_t_select.

  DATA: ls_select TYPE srsc_s_select.

  DATA: lt_select TYPE srsc_t_select.

  DATA: ls_return TYPE bapiret2.

  DATA: lv_tabix  TYPE sytabix.

  lt_select = ct_select.

  SORT lt_select BY fieldnm.
  DELETE lt_select WHERE fieldnm NE zcon_gc_fn_budat.

  DESCRIBE TABLE lt_select.
  IF sy-tfill > 1.
    MESSAGE e113(zstocktransit) WITH zcon_gc_fn_budat
                      RAISING error_passed_to_mess_handler.
* Selection parameter & only allows one value
  ENDIF.

  READ TABLE lt_select INTO ls_select
                        WITH KEY fieldnm = zcon_gc_fn_budat.

  IF sy-subrc NE 0.
    MESSAGE e114(zstocktransit) WITH zcon_gc_fn_budat
                      RAISING error_passed_to_mess_handler.
* Parameter & is mandatory
  ENDIF.

  IF NOT ls_select-low IS INITIAL AND NOT ls_select-high IS INITIAL
    AND ls_select-low NE ls_select-high .
    MESSAGE e115(zstocktransit) WITH zcon_gc_fn_budat.
* For parameter & no intervall is allowed
  ENDIF.

  READ TABLE ct_select INTO ls_select
                        WITH KEY fieldnm = zcon_gc_fn_budat.

  lv_tabix = sy-tabix.

  IF NOT ls_select-high IS INITIAL.
    CLEAR: ls_select-high.
  ENDIF.

  ls_select-option = 'LE'.

  MODIFY ct_select FROM ls_select INDEX lv_tabix.

ENDFORM.                    " budat_validate
```

g) Include LZZ_STOCK_TRANSITF0C

```
*&---------------------------------------------------------------------
```

```
*
*&  Include           LZZ_STOCK_TRANSITF0C
*
*&---------------------------------------------------------------------
*

FORM check_relevance USING    it_ekbe TYPE SORTED TABLE
                              iv_ebeln
                              iv_ebelp
                      CHANGING cv_relevant
                               cv_wamng
                               cv_wemng.

  DATA: ls_ekbe TYPE ekbe,
        ls_cum  TYPE zcum_struc_transit.

  DATA: lv_start       TYPE sy-tabix,
        lv_ebelnfield  TYPE fieldname VALUE 'EBELN',
        lv_ebelpfield  TYPE fieldname VALUE 'EBELP',
        lv_wamng       TYPE menge_d,
        lv_wemng       TYPE menge_d.

  READ TABLE it_ekbe TRANSPORTING NO FIELDS
                                  WITH KEY (lv_ebelnfield) = iv_ebeln
                                           (lv_ebelpfield) = iv_ebelp
                                           BINARY SEARCH.

  IF sy-subrc NE 0.
    CLEAR: cv_relevant.
    EXIT.
  ELSE.
    lv_start = sy-tabix.
  ENDIF.

  LOOP AT it_ekbe INTO ls_ekbe FROM lv_start.

    IF ls_ekbe-ebeln NE iv_ebeln OR
       ls_ekbe-ebelp NE iv_ebelp.
      EXIT.
    ENDIF.

    MOVE: ls_ekbe-ebeln TO ls_cum-ebeln,
          ls_ekbe-ebelp TO ls_cum-ebelp.

    CASE ls_ekbe-vgabe.

      WHEN '1'.
* goods receipt
        IF ls_ekbe-shkzg EQ 'S'.
* normal goods receipt
          ADD ls_ekbe-menge TO ls_cum-wemng.
        ELSE.
* reversal
          ls_ekbe-menge = ls_ekbe-menge * -1.
          ADD ls_ekbe-menge TO ls_cum-wemng.
        ENDIF.
      WHEN '6'.
* goods issue
        IF ls_ekbe-shkzg EQ 'H'.
* normal goods issue
          ADD ls_ekbe-menge TO ls_cum-wamng.
```

```
        ELSE.
* reversal
          ls_ekbe-menge = ls_ekbe-menge * -1.
          ADD ls_ekbe-menge TO ls_cum-wamng.
        ENDIF.

    ENDCASE.

  ENDLOOP.

  IF ls_cum-wamng <> ls_cum-wemng.

    cv_relevant = 'X'.
    cv_wamng = ls_cum-wamng.
    cv_wemng = ls_cum-wemng.
  ELSE.

    CLEAR: cv_relevant.

  ENDIF.

ENDFORM.                    " check_relevence
```

h) Include LZZ_STOCK_TRANSITF0D

```
*&---------------------------------------------------------------------
*
*&  Include           LZZ_STOCK_TRANSITF0D
*
*&---------------------------------------------------------------------
*

FORM data_selection_transit_date USING    iv_maxsize
                                    ir_material    TYPE STANDARD TABLE
                                    ir_plant       TYPE STANDARD TABLE
                                    ir_pstyp       TYPE STANDARD TABLE
                                    ir_budat       TYPE STANDARD TABLE
                                    it_sel_fields  TYPE STANDARD TABLE
                            CHANGING ct_data       TYPE STANDARD
TABLE.

  DATA: lr_plant TYPE RANGE OF werks_d,
        lr_ebeln TYPE RANGE OF ebeln,
        lr_ebelp TYPE RANGE OF ebelp,
        lr_etenr TYPE RANGE OF etenr.

  DATA: lt_data TYPE STANDARD TABLE OF zstock_transit_sel,
        lt_mara TYPE SORTED TABLE OF mara WITH UNIQUE KEY matnr,
        lt_fields       TYPE srsc_t_fields,
        lt_transit_data TYPE STANDARD TABLE OF zstock_transit.

  DATA: ls_data         TYPE zstock_transit_sel,
        ls_mara         TYPE mara,
        ls_t001         TYPE t001,
        ls_transit_data TYPE zstock_transit.

  DATA: lv_lines   TYPE sytfill,
        lv_maxsize TYPE rsmaxsize.

  STATICS: ls_max_transit_data TYPE zstock_transit_sel,
           lt_sel_fields       TYPE STANDARD TABLE OF fieldname,
```

```
          lt_ekbe             TYPE SORTED TABLE OF ekbe
                                     WITH UNIQUE KEY ebeln
                                                     ebelp
                                                     zekkn
                                                     vgabe
                                                     gjahr
                                                     belnr
                                                     buzei,
          lt_t001             TYPE SORTED TABLE OF t001
                                       WITH UNIQUE KEY bukrs.

STATICS: lv_datapakid_counter  TYPE sytabix,
         lv_cursor             TYPE cursor,
         lv_tabix              TYPE sytabix,
         lv_wamng              TYPE menge_d,
         lv_wemng              TYPE menge_d,
         lv_relevant           TYPE char1.

DATA:    lv_test               TYPE char1 VALUE 'X'.

DATA:    lv_ebeln_field        TYPE fieldname VALUE 'EBELN',
         lv_ebelp_field        TYPE fieldname VALUE 'EBELP',
         lv_budat              TYPE datum,
         lv_etenr_field        TYPE fieldname VALUE 'ETENR'.

DATA:    lp_data TYPE REF TO data.

FIELD-SYMBOLS: <ls_budat> TYPE ANY,
               <lv_low>   TYPE ANY.

CREATE DATA lp_data LIKE LINE OF ir_budat.
ASSIGN lp_data->* TO <ls_budat>.

ASSIGN COMPONENT 'LOW' OF STRUCTURE <ls_budat> TO <lv_low>.

IF lt_sel_fields IS INITIAL.

* adjust selected fields to your requirements

  SELECT fieldname FROM dd03l INTO TABLE lt_fields
                              WHERE tabname  = 'ZSTOCK_TRANSIT_SEL'
                              AND   comptype <> 'S'.

  PERFORM field_list_build USING     lt_fields
                           CHANGING  lt_sel_fields.

  SELECT * FROM t001 INTO TABLE lt_t001.

ENDIF.

IF lv_datapakid_counter IS INITIAL.
  OPEN CURSOR WITH HOLD lv_cursor FOR
  SELECT (lt_sel_fields)
       FROM mb_mdbs INNER JOIN ekko
       ON mb_mdbs~ebeln = ekko~ebeln
       WHERE mb_mdbs~ebeln IN lr_ebeln
       AND   mb_mdbs~ebelp IN lr_ebelp
       AND   mb_mdbs~etenr IN lr_etenr
       AND   mb_mdbs~matnr IN ir_material
       AND   mb_mdbs~werks IN ir_plant
       AND   mb_mdbs~pstyp IN ir_pstyp
```

```
          AND   mb_mdbs~loekz EQ space
*         and   mb_mdbs~elikz eq space
          AND   mb_mdbs~bstyp EQ 'F'
          AND   ekko~reswk <> space.

      IF sy-subrc NE 0.
        MESSAGE e103(zstocktransit) RAISING no_more_data.
*         No data found
      ENDIF.
    ENDIF.
    FETCH NEXT CURSOR lv_cursor
               INTO CORRESPONDING FIELDS
               OF TABLE lt_data
               PACKAGE SIZE iv_maxsize.

    IF sy-subrc <> 0.
      CLOSE CURSOR lv_cursor.
      MESSAGE e103(zstocktransit) RAISING no_more_data.
    ENDIF.
    lv_datapakid_counter = lv_datapakid_counter + 1.

* adjust selected fields to your requirements

    SELECT * FROM mara INTO TABLE lt_mara FOR ALL ENTRIES IN lt_data
                       WHERE matnr EQ lt_data-matnr.

    SELECT * FROM ekbe INTO TABLE lt_ekbe FOR ALL ENTRIES IN lt_data
                                  WHERE ebeln EQ lt_data-ebeln
                                  AND   ebelp EQ lt_data-ebelp
                                  AND  ( vgabe EQ '1'
                                  OR vgabe EQ '6' )
                                  AND budat IN ir_budat.

    READ TABLE ir_budat INTO <ls_budat> INDEX 1.

    lv_budat = <lv_low>.

    LOOP AT lt_data INTO ls_data.

      lv_tabix = sy-tabix.
      CLEAR: lv_relevant.

      PERFORM check_relevance USING     lt_ekbe
                                        ls_data-ebeln
                                        ls_data-ebelp
                              CHANGING  lv_relevant
                                        lv_wamng
                                        lv_wemng.

      IF lv_relevant IS INITIAL.

        DELETE lt_data INDEX lv_tabix.
        CONTINUE.

      ENDIF.

      READ TABLE lt_mara INTO ls_mara WITH KEY matnr = ls_data-matnr
                                               BINARY SEARCH.

      READ TABLE lt_t001 INTO ls_t001 WITH KEY bukrs = ls_data-bukrs
                                               BINARY SEARCH.
```

```
    MOVE-CORRESPONDING ls_data TO ls_transit_data.

    ls_transit_data-budat = lv_budat.
    ls_transit_data-menge = lv_wamng - lv_wemng.

    IF NOT ls_data-retpo IS INITIAL.
      ls_transit_data-werks = ls_data-reswk.
      IF ls_transit_data-menge < 0.
        ls_transit_data-menge = ls_transit_data-menge * -1.
      ENDIF.
    ENDIF.

* adjustment NETWR to quantity in transit

    ls_data-netwr = ls_transit_data-menge *
                                ls_data-netwr / ls_data-bstmg.

    ls_transit_data-menge = ls_transit_data-menge *
                            ls_data-umrez / ls_data-umren.

    ls_transit_data-meins = ls_mara-meins.
    ls_transit_data-attyp = ls_mara-attyp.

    ls_transit_data-bsttyp = 'A'.
    IF ls_t001-waers NE ls_data-waers.

      CALL FUNCTION 'CONVERT_TO_LOCAL_CURRENCY'
        EXPORTING
          date             = sy-datum
          foreign_amount   = ls_data-netwr
          foreign_currency = ls_data-waers
          local_currency   = ls_t001-waers
          rate             = ls_data-wkurs
        IMPORTING
          local_amount     = ls_transit_data-dmbtr.

      ls_transit_data-hwaer  = ls_t001-waers.

    ELSE.
      ls_transit_data-dmbtr  = ls_data-netwr.
      ls_transit_data-hwaer  = ls_data-waers.

    ENDIF.

    COLLECT ls_transit_data INTO lt_transit_data.

  ENDLOOP.

  ct_data = lt_transit_data.

ENDFORM.                    " data_selection_transit
```

i) Include LZZ_STOCK_TRANSITF0F

```
*&---------------------------------------------------------------------
*
*&  Include           LZZ_STOCK_TRANSITF0F
*
*&---------------------------------------------------------------------
*
```

```
FORM field_list_build  USING    it_fields    TYPE srsc_t_fields
                       CHANGING ct_fields TYPE ANY TABLE.

  DATA: lr_fields TYPE RANGE OF fieldname.

  DATA: lt_fields TYPE srsc_t_fields.

  DATA: ls_fields TYPE srsc_s_fields.

  LOOP AT it_fields INTO ls_fields.
    PERFORM build_range USING    zcon_gc_sign_include
                                 zcon_gc_option_equal
                                 ls_fields-fieldnm
                                 space
                        CHANGING lr_fields.
  ENDLOOP.

  PERFORM get_fields USING    'MB_MDBS'
                     CHANGING lr_fields
                              lt_fields.

  PERFORM get_fields USING    'EKKO'
                     CHANGING lr_fields
                              lt_fields.

  SORT lt_fields BY fieldnm.
  DELETE ADJACENT DUPLICATES FROM lt_fields COMPARING fieldnm.
  ct_fields = lt_fields.

ENDFORM.                    " field_list_build
```

j) Include LZZ_STOCK_TRANSITF0G

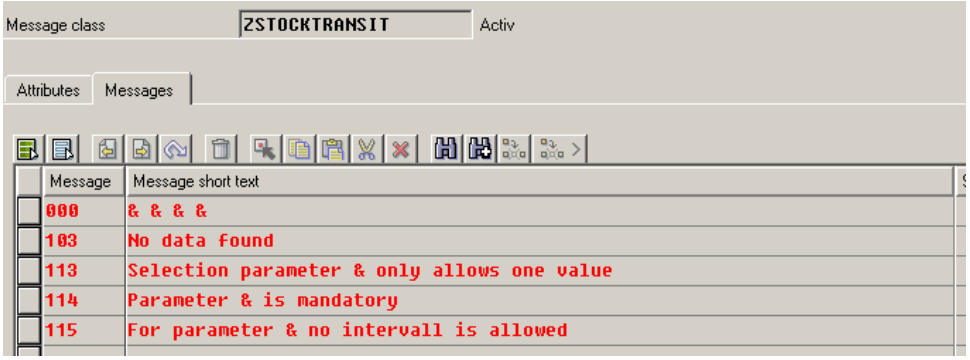
```
*&---------------------------------------------------------------------
*
*&  Include           LZZ_STOCK_TRANSITF0G
*
*&---------------------------------------------------------------------
*

FORM get_fields USING    iv_tabname
                CHANGING cr_fields   TYPE STANDARD TABLE
                         ct_fields   TYPE srsc_t_fields.

  DATA: lt_fieldname TYPE STANDARD TABLE OF fieldname.
  DATA: ls_field     TYPE srsc_s_fields.
  DATA: lv_fieldname TYPE fieldname.
  DATA: lt_fields    TYPE srsc_t_fields.
  DATA: lp_data TYPE REF TO data.

  FIELD-SYMBOLS: <ls_range> TYPE ANY,
                 <lv_low>   TYPE ANY.

  CREATE DATA lp_data LIKE LINE OF cr_fields.
  ASSIGN lp_data->* TO <ls_range>.

  ASSIGN COMPONENT 'LOW' OF STRUCTURE <ls_range> TO <lv_low>.

  SELECT fieldname FROM dd03l APPENDING TABLE ct_fields
                   WHERE tabname   = iv_tabname
```

```
                    AND   fieldname IN cr_fields.

  LOOP AT cr_fields INTO <ls_range>.

    READ TABLE ct_fields INTO ls_field WITH KEY fieldnm = <lv_low>.
    IF sy-subrc EQ 0.
      CONCATENATE iv_tabname '~' ls_field-fieldnm INTO ls_field-
fieldnm.
      MODIFY ct_fields FROM ls_field INDEX sy-tabix.
      DELETE TABLE cr_fields FROM <ls_range>.
    ENDIF.
  ENDLOOP.


ENDFORM.                     " get_fields
```

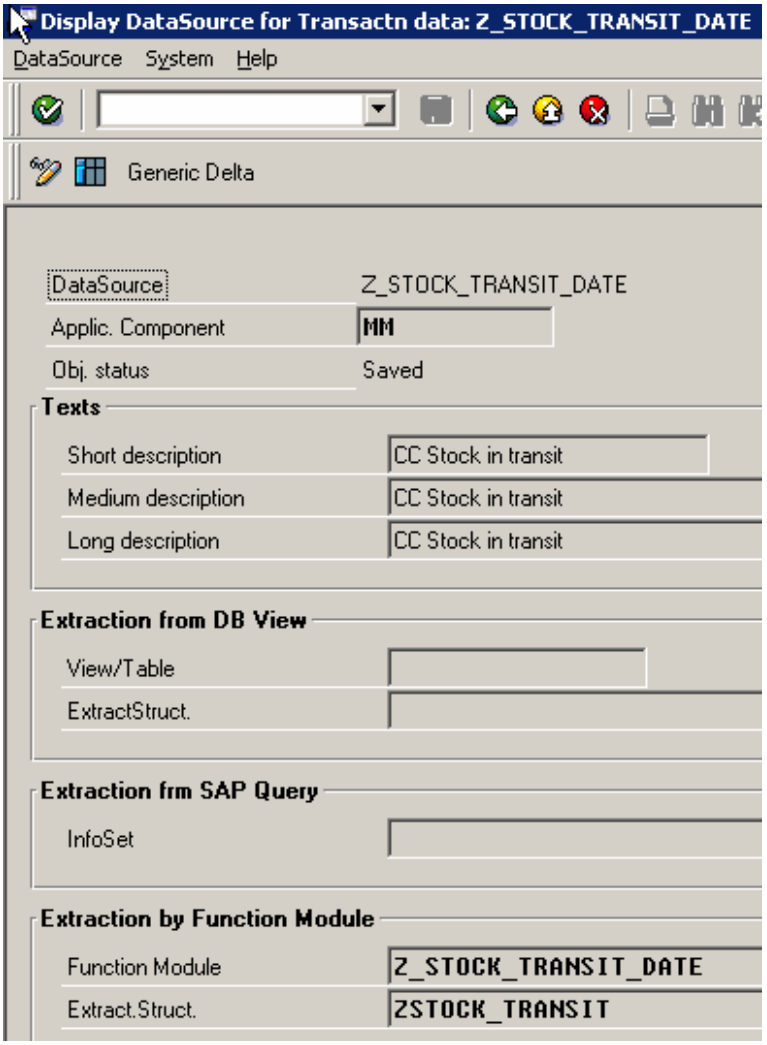
k) Include LZZ_STOCK_TRANSITF0R

```
*&---------------------------------------------------------------------
*
*&  Include           LZZ_STOCK_TRANSITF0R
*
*&---------------------------------------------------------------------
*

FORM range_fill USING    iv_selfield
                         it_seltab  TYPE srsc_t_select
                CHANGING er_range TYPE STANDARD TABLE.


  DATA: lp_data   TYPE REF TO data.
  DATA: ls_seltab TYPE srsc_s_select.
  FIELD-SYMBOLS: <ls_line> TYPE ANY.

  CREATE DATA lp_data LIKE LINE OF er_range.
  ASSIGN lp_data->* TO <ls_line>.

  LOOP AT it_seltab INTO ls_seltab
                         WHERE fieldnm = iv_selfield.
    PERFORM build_range USING ls_seltab-sign
                              ls_seltab-option
                              ls_seltab-low
                              ls_seltab-high
                        CHANGING er_range.
  ENDLOOP.


ENDFORM.                     " range_fill
```

3. Message class ZSTOCKTRANSIT



3. Creation of DataSource with RSO2

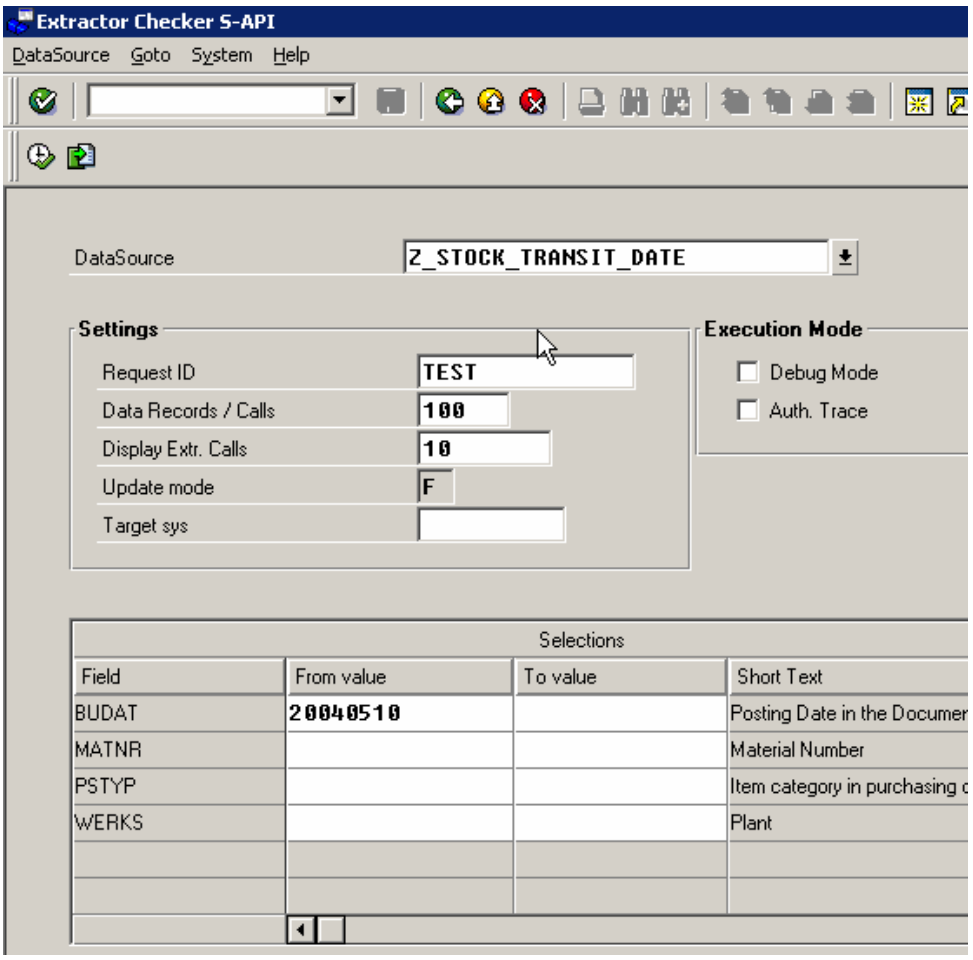


Field Name	Short text	Selection	Hide field	Inversion	Field only Kno
ATTYP	Material category	☐	☐	☐	☐
BSTAUS	Stock values in the Logistics Information System	☐	☐	☐	☐
BSTTYP	Stock categories in the Logistics Information System	☐	☐	☐	☐
BUDAT	Posting Date in the Document	☑	☐	☐	☐
BUKRS	Company Code	☐	☐	☐	☐
DMBTR	Amount in local currency	☐	☐	☐	☐
EBELN	Purchasing Document Number	☐	☐	☐	☐
EBELP	Item Number of Purchasing Document	☐	☐	☐	☐
ETENR	Schedule line	☐	☐	☐	☐
HWAER	Currency Key	☐	☐	☐	☐
MATNR	Material Number	☑	☐	☐	☐
MEINS	Base Unit of Measure	☐	☐	☐	☐
MENGE	Quantity	☐	☐	☐	☐
MTART	Material Type	☐	☐	☐	☐
PRICE_DATE	Field of type DATS	☐	☐	☐	☐
PSTYP	Item category in purchasing document	☑	☐	☐	☐
RESWK	Supplying (issuing) plant in case of stock transport orde	☐	☐	☐	☐

Field Name	Short text	Selection	Hide field	Inversion	Field only Kno
SATNR	Cross-Plant Configurable Material	☐	☐	☐	☐
VKORG	Sales Organization	☐	☐	☐	☐
VLFKZ	Plant category	☐	☐	☐	☐
VTWEG	Distribution Channel	☐	☐	☐	☐
WERKS	Plant	☑	☐	☐	☐

4. Data Source testing (RSA3) and comparions with MB5T

1. Selection Screen RSA3 (selection of posting date is mandatory!)



2. Test result (material R100000)[2]

[2] Please note that the company code is the company code from the supplying plant. Also the quantity and value belongs to the supplying plant/company code. The value is determined by the purchase order value. In the inter company stock transfer (coverd by 2LIS_03_BF) the value is determined with the valuation price of the receiving plant.

DataSource Z_STOCK_TRANSIT_DATE
Structure ZSTOCK_TRANSIT

Material category	00
Stock category LIS	A
Posting Date	10.05.2004
Company Code	R300
Amt.in loc.cur.	10,60
Purchasing Document	4500004885
Item	10
Schedule line number	1
Currency	USD
Material	R100000
Base Unit of Measure	PC
Quantity	20,000
Item category	7
Supplying plant	R300
Plant	R314

3. Result with TAC MB5T

Display of Stock in Transit

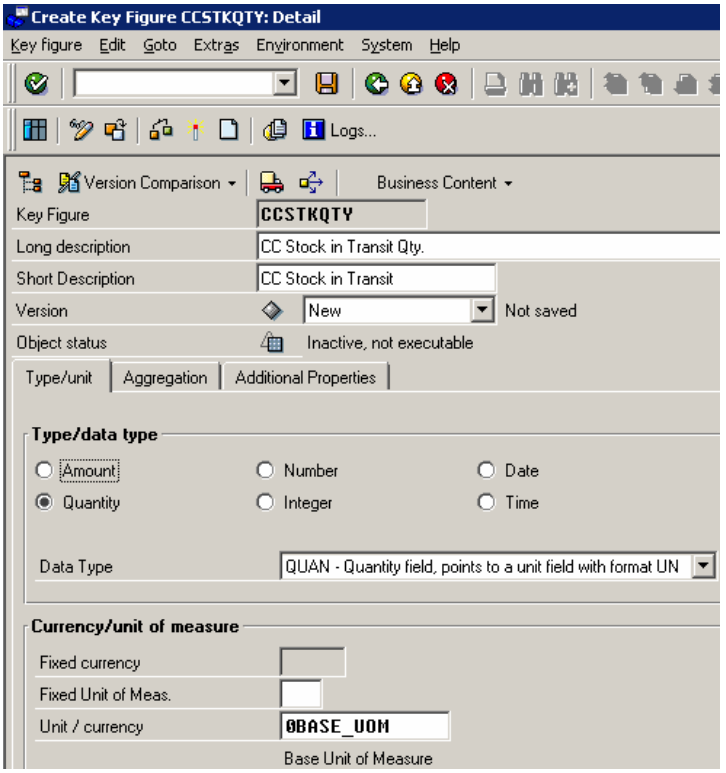
List Edit Goto Settings Environment System Help

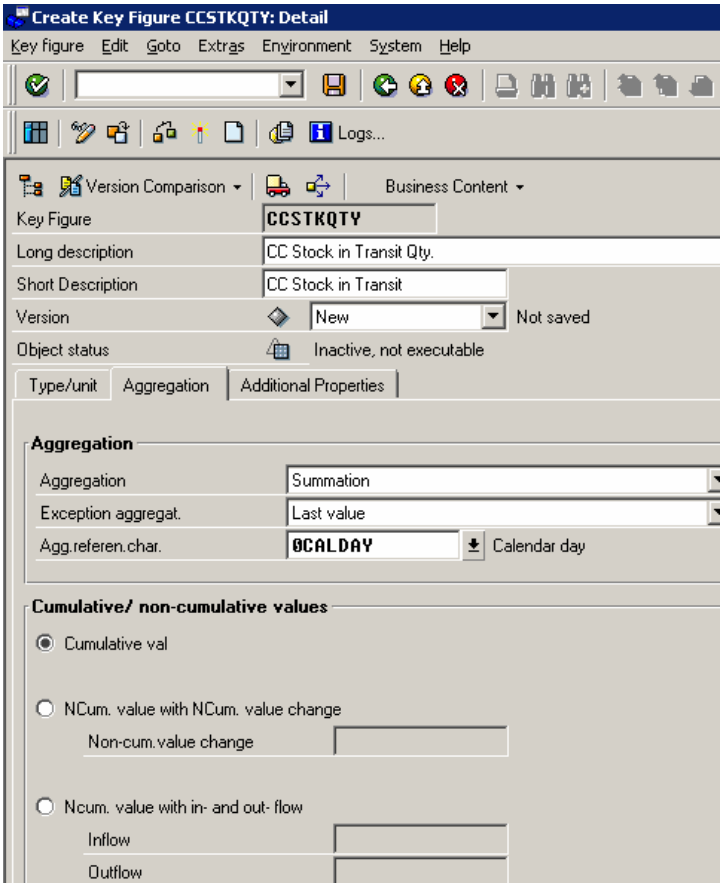
Material	Material Description				Plnt Name 1
Purch.Doc.	Item SPlt S	Quantity	BUn	Amt.in loc.cur.	Crcy Order qu
R100000	Yoghurt all natural				R110 SB Warenhaus R1
4500010291	10 R100	13.770	PC	0,00	EUR
R100000	Yoghurt all natural				R310 GM Store R310
4500009935	10 R300	3.840	PC	0,00	USD
R100000	Yoghurt all natural				R314 GM Store R314
4500004885	10 R300	20	PC	0,00	USD

2. Steps in BW

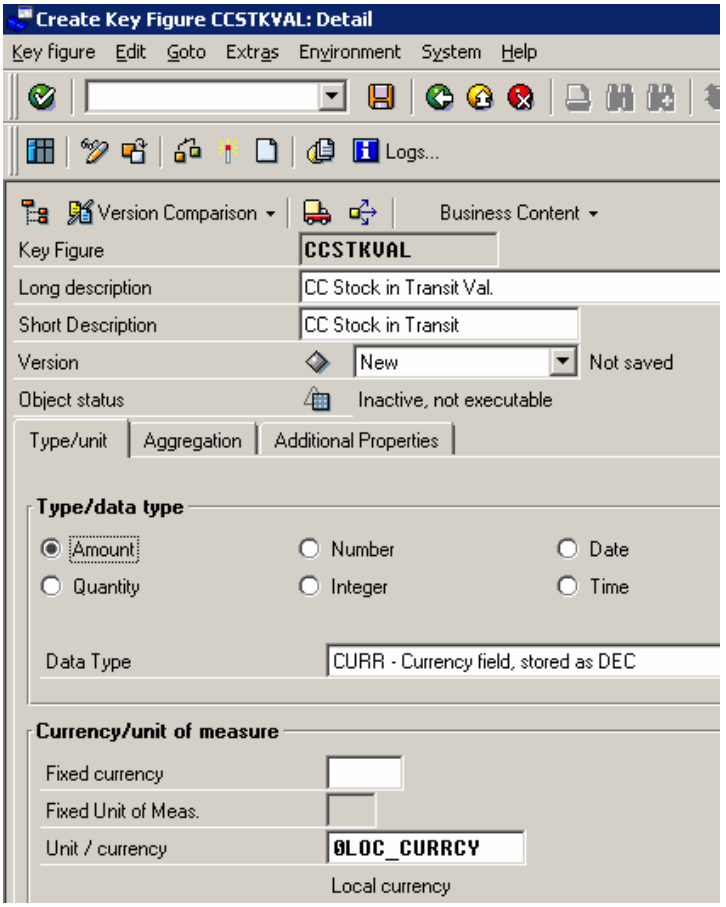
1. Creation of missing InfoObjects

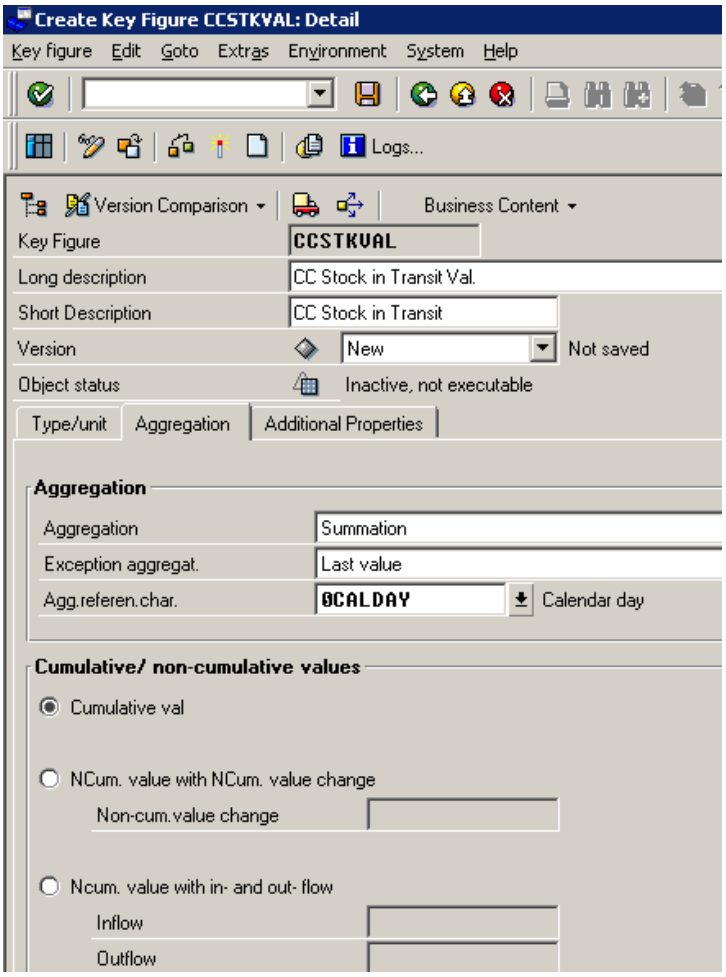
1. CCSTKQTY (CC Stock in Transit Qty.)





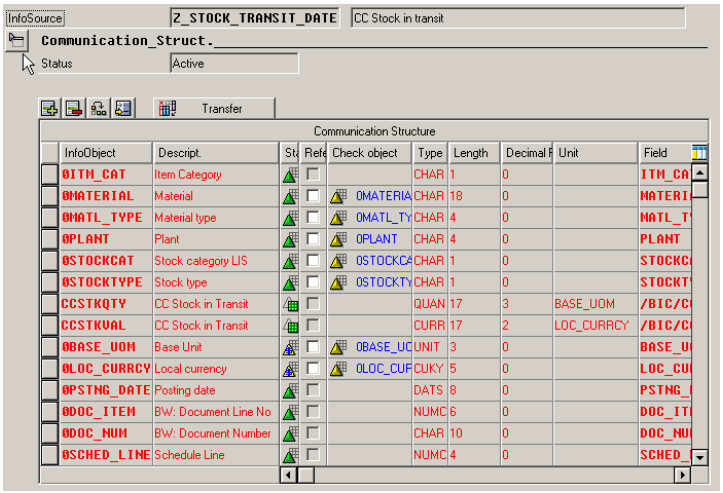
1. CCSTKVAL (CC Stock in Transit Val.)

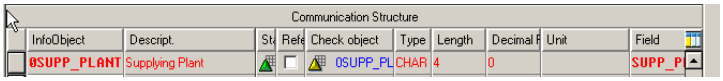

Create Key Figure CCSTKVAL: Detail
Key figure Edit Goto Extras Environment System Help
Logs...
Version Comparison
Business Content
Key Figure
CCSTKVAL
Long description
CC Stock in Transit Val.
Short Description
CC Stock in Transit
Version
New
Not saved
Object status
Inactive, not executable
Type/unit
Aggregation
Additional Properties
Type/data type
Amount
Number
Date
Quantity
Integer
Time
Data Type
CURR - Currency field, stored as DEC
Currency/unit of measure
Fixed currency
Fixed Unit of Meas.
Unit / currency
0LOC_CURRCY
Local currency



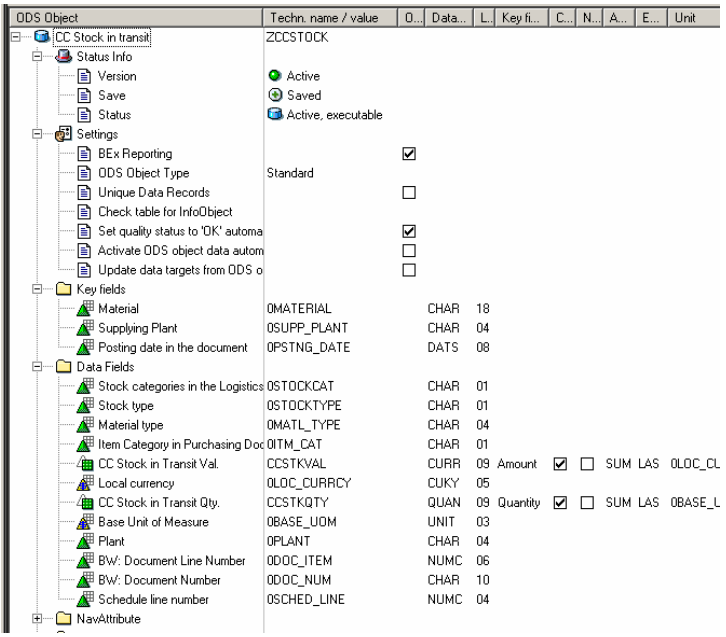
2. Creation of InfoSource, ODS and update rules

1. InfoSource Z_STOCK_TRANSIT_DATE





2. ODS ZCCSTOCK

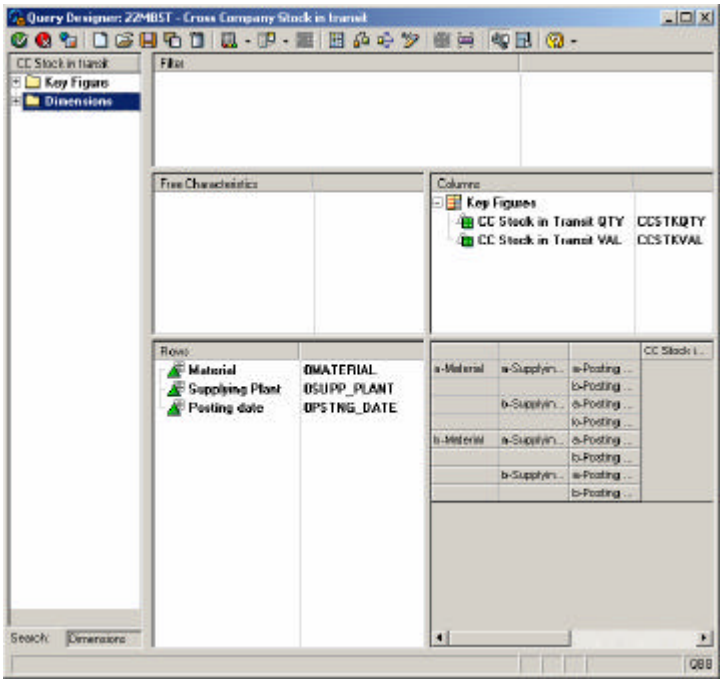


ODS Object	Techn. name / value	O...	Data...	L...	Key fi...	C...	N...	A...	E...	Unit
CC Stock in transit	ZCCSTOCK									
Status Info										
Version	Active									
Save	Saved									
Status	Active, executable									
Settings										
BEx Reporting		☑								
ODS Object Type	Standard									
Unique Data Records		☐								
Check table for InfoObject										
Set quality status to 'OK' automa		☑								
Activate ODS object data autom		☐								
Update data targets from ODS o		☐								
Key fields										
Material	0MATERIAL		CHAR	18						
Supplying Plant	0SUPP_PLANT		CHAR	04						
Posting date in the document	0PSTNG_DATE		DATS	08						
Data Fields										
Stock categories in the Logistics	0STOCKCAT		CHAR	01						
Stock type	0STOCKTYPE		CHAR	01						
Material type	0MATL_TYPE		CHAR	04						
Item Category in Purchasing Doc	0ITM_CAT		CHAR	01						
CC Stock in Transit Val.	CCSTKVAL		CURR	09	Amount	☑	☐	SUM	LAS	0LOC_CL
Local currency	0LOC_CURRCY		CUKY	05						
CC Stock in Transit Qty.	CCSTKQTY		QUAN	09	Quantity	☑	☐	SUM	LAS	0BASE_L
Base Unit of Measure	0BASE_UOM		UNIT	03						
Plant	0PLANT		CHAR	04						
BW: Document Line Number	0DOC_ITEM		NUMC	06						
BW: Document Number	0DOC_NUM		CHAR	10						
Schedule line number	0SCHED_LINE		NUMC	04						
NavAttribute										

3. Update rules

There is a complete 1:1 Mapping. Please set the key figures to "Addition"

3. Creation of a sample query

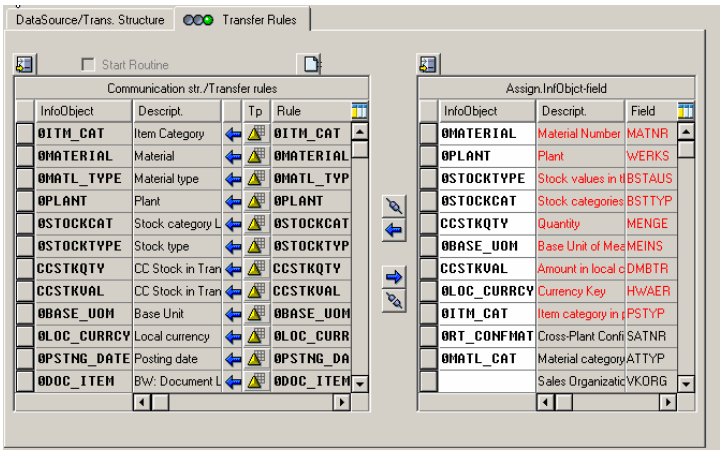


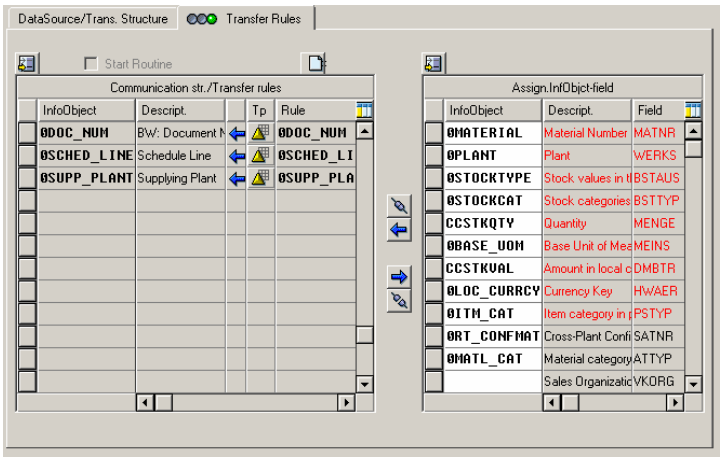
4. Date Source replication; creation of transfer rules and data extraction from R/3

1. Replication of DataSource Z_STOCK_TRANSIT_DATE

2. Creation of transfer rules

Not every field gets assigned from the DataSource, if further fields for reporting are required, enhancements or modifications can be done easily.





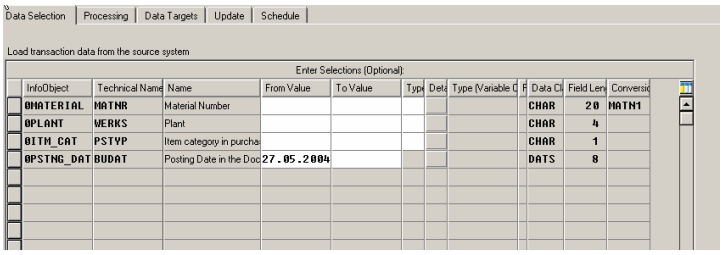
3. Data extraction - Scheduler

At least the selection criteria posting date has to be provided. Otherwhise the extractor will give an error if no selection criteria are handed over.

Please note that by restricting 0ITM_CAT with “0” only the cross company stock in transit gets extracted. Restrictions with “7” gives the inter company stock in transit back. No selections restrict the complete stock in

transit.[3]

As we have here a snapshot scenario update mode "Full Update" is the only supported one. This means that no duplicated uploads with the same posting date should be executed.



5. Query testing and comparison with MB5T

1. Query ZZMB5T

[3] Please note that in same cases this restriction doesn't help. See comment

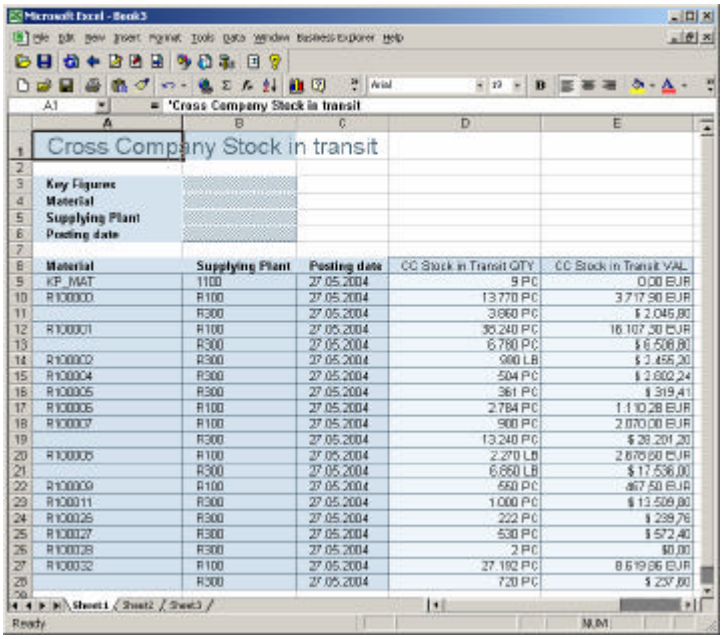
```
* If the external range XPSTYP is filled, try to guess what types
* of PO's should be selected. This is just for backwards compatibility
* with callers which do not use the separated flags.
* Please note that with cross-system movements, PO's with UB-logic
* (formerly PSTYP = 7) can also have PSTYP = 0. Therefore, this
* criteria is not usable any more.
  IF NOT xpstyp[] IS INITIAL.
    IF '0' IN xpstyp OR '3' IN xpstyp.
      i_cross_company = 'X'.
    ENDIF.
    IF '7' IN xpstyp.
      i_non_cross_company = 'X'.
    ENDIF.

  ENDIF.
```

in function module "MB_ADD_TRANSFER_QUANTITY" that is used in transaction MB5T.

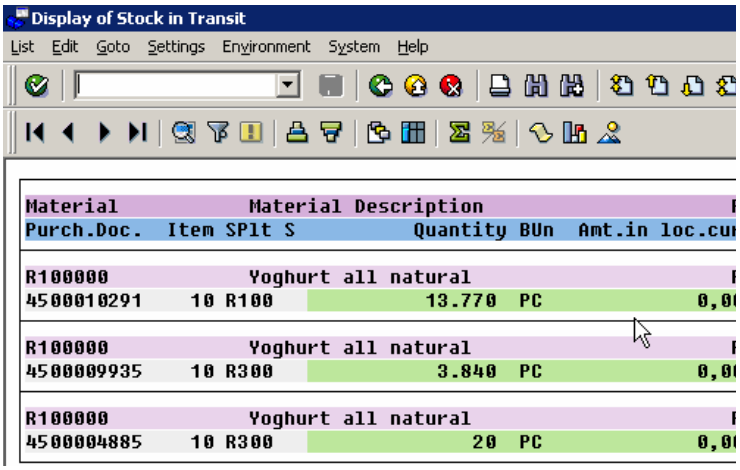
In such cases the similar logic

```
* Delete if not requested
  IF ( l_bsakz = 'T'   AND i_non_cross_company IS INITIAL ) OR
     ( l_bsakz = space AND i_cross_company     IS INITIAL ).
    DELETE xmdbs.
    CONTINUE.

  ENDIF.
```

from this function module has to be implemented in the extractor.



2. Transaction MB5T



6. Creation of a Multi Provider and Multi Provider Query

Together with InfoCube 0IC_C03 the defined ODS object can be part of a MultiProvider. The most common objects like material, plant and posting date can be defined as matching characteristics.

7. Query testing of consolidated stock reporting

With this MultiProvider consolidated stock reporting can be achieved.

4 Additional Information

This scenarion has been tested with 3.0B SP20 and R/3 Enterprise 4.7 SP0 but should also work with any BW >= 2.0B and R/3 >= 4.0B combination.