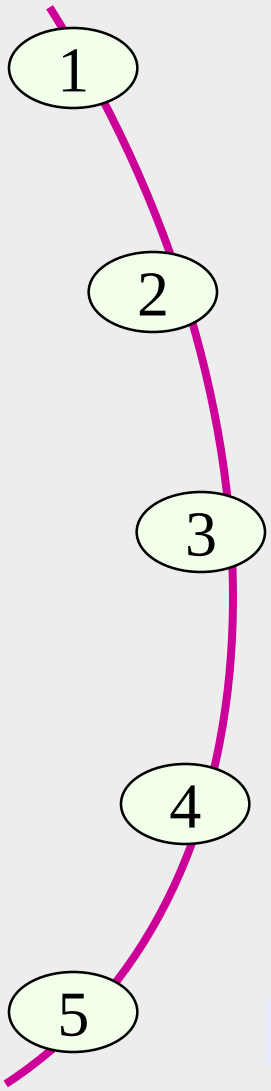
SAP BW

Lesson 11: ABAP, Open-hub Overview

Capgemini

BI related ABAP Overview & open-hub



Role of ABAP in BI

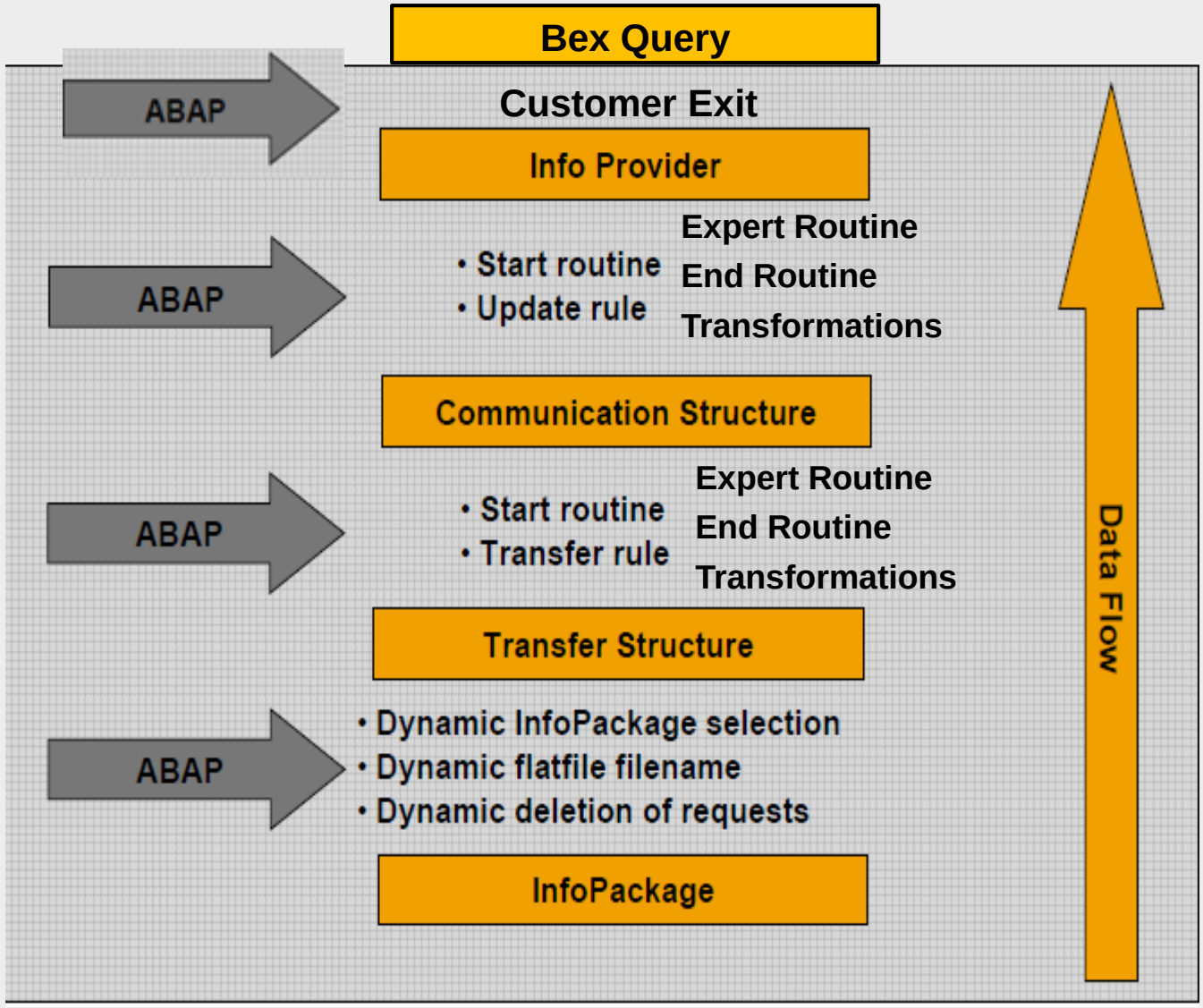
DataSource Enhancements

Query level Bex user Exits

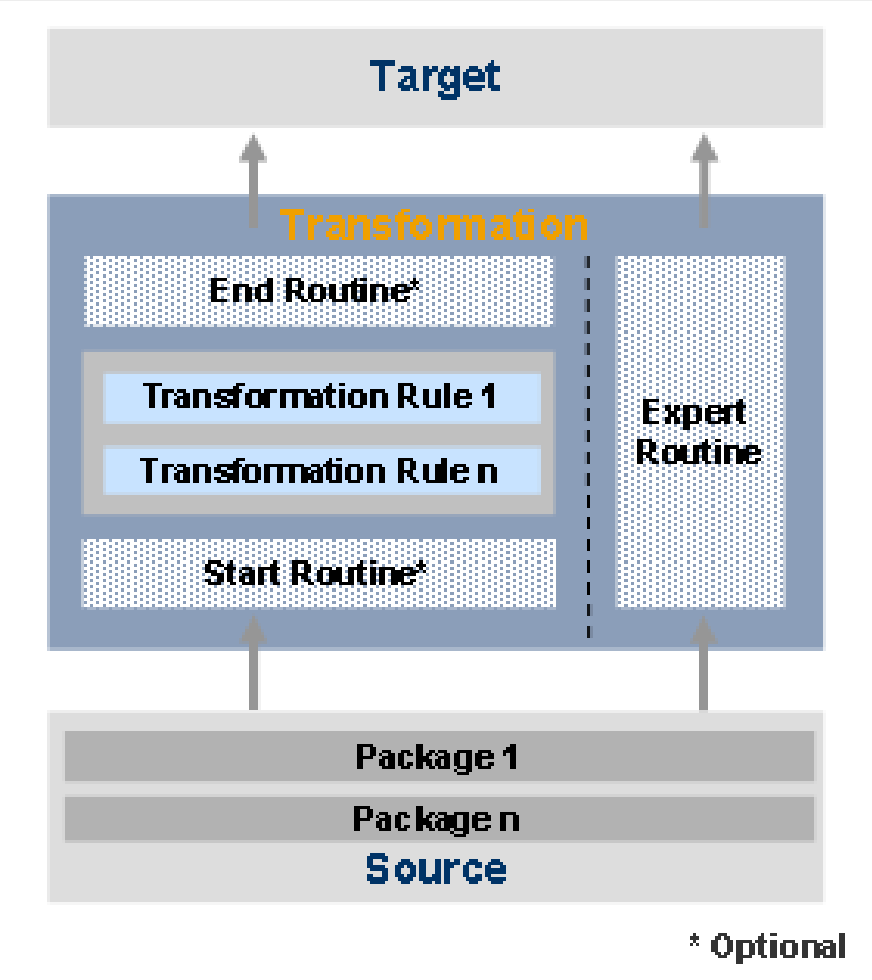
Data Staging

Introduction to the Open Hub Service

1. Role of ABAP in BI – 3.x



1.1 BI - Transformation



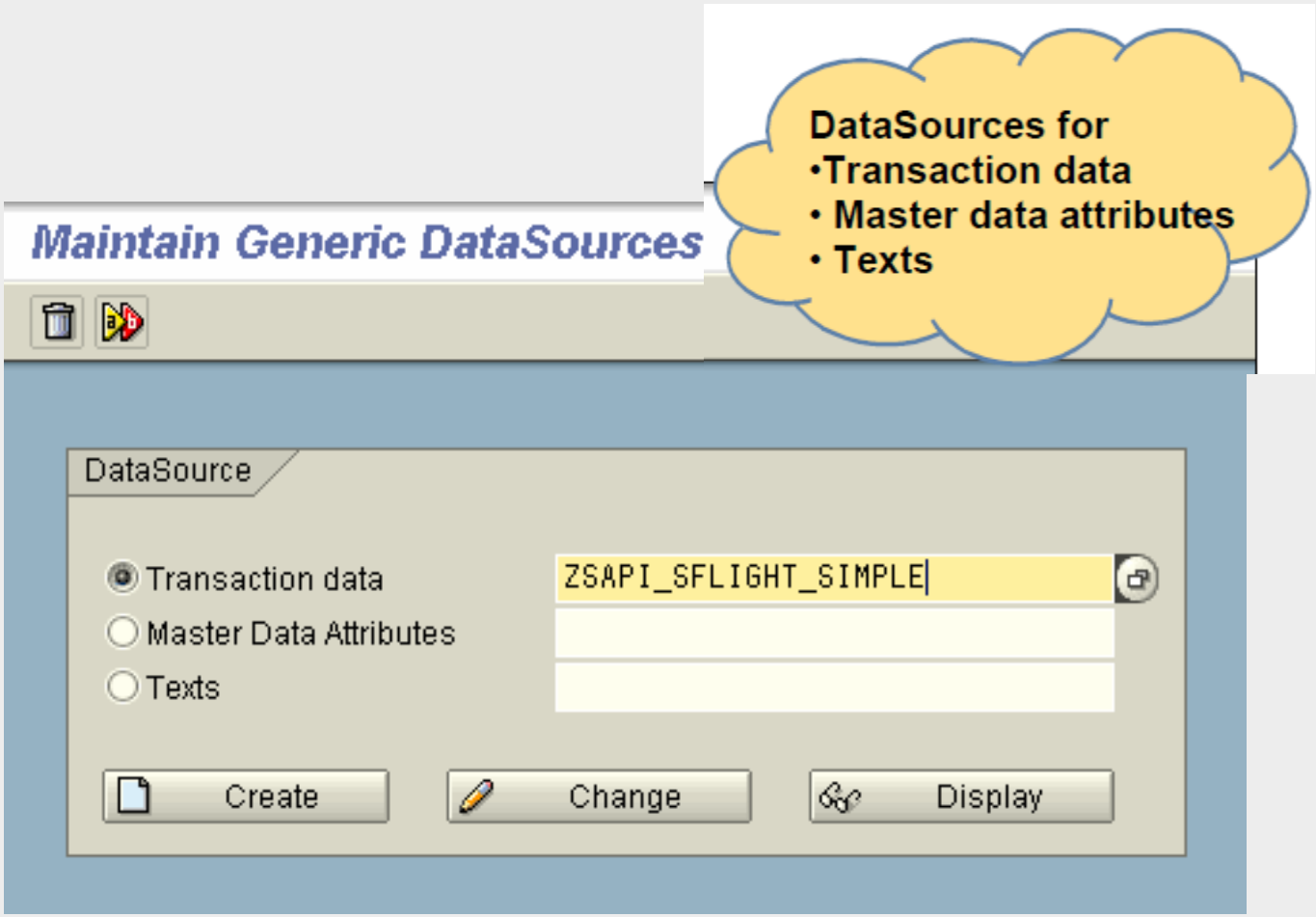
2. DataSource Enhancements

- Creating generic Data Sources using function modules for Data Extraction
- Enhancing Data Sources for transaction data, master data attributes, and texts
- Using the Service API function enhancement to enrich data
- Overview of the options available for enhancing data in the source system

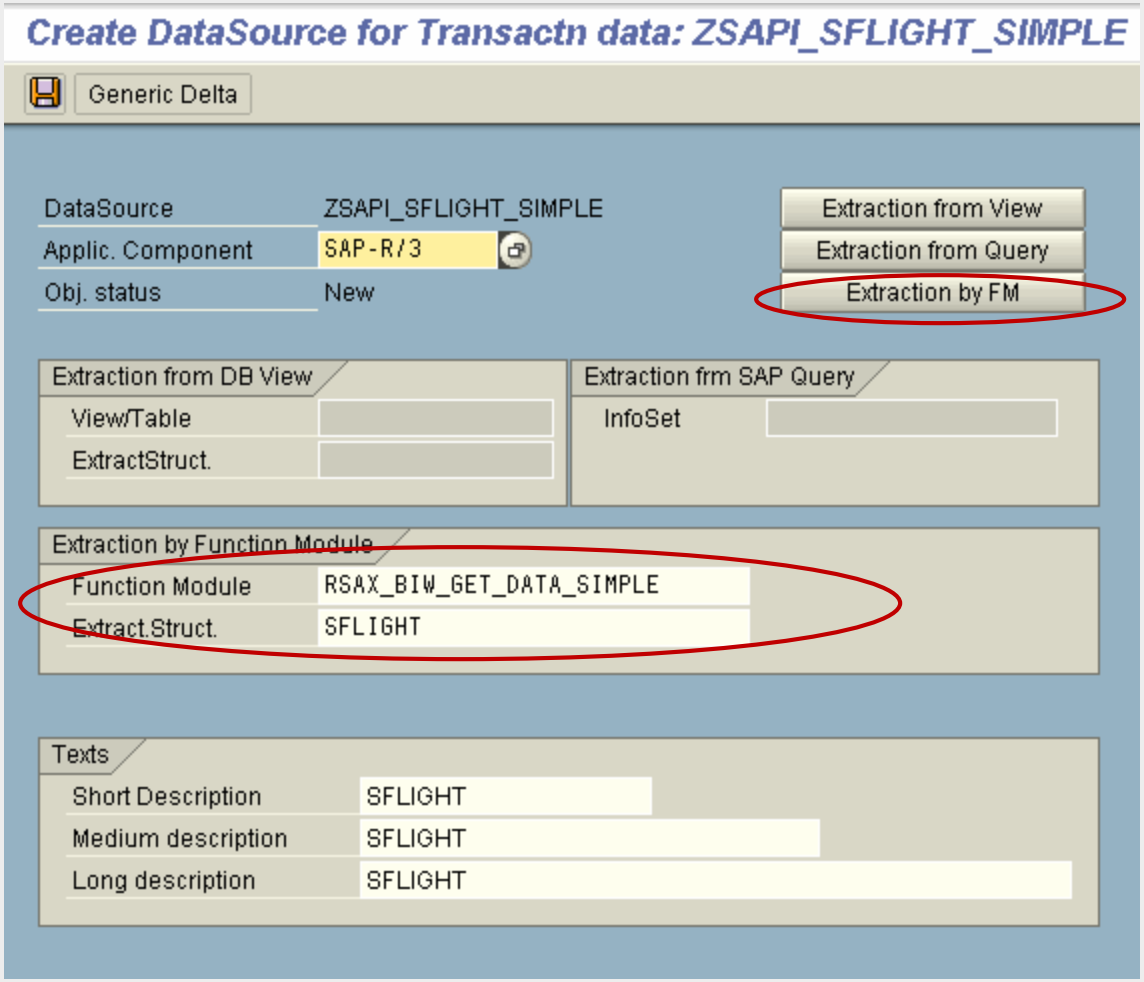
Using the Tools for Generic Data Extraction

- When should we use the generic data extraction tools to create a DataSource?
- Business Content does not include a DataSource for your application
- You want to implement a delta method on your generic DataSource that cannot be implemented by using the generic delta functionality (timestamp, date ...)
- The application does not allow you to create additional application specific generic extractors (CO-PA, FI-SL, LIS).
- You use your own programs in the SAP system to populate your own tables
- You have to extract data from several DB tables when:
- Using a view is not possible because of insufficient JOIN
- Only some fields are relevant to these tables
- Data has to be enriched with information not available in the BW system

Tools for Generic Data Extraction: RSO2



Creating DataSources for Generic Data Extraction



Enhancing Business Content DataSources

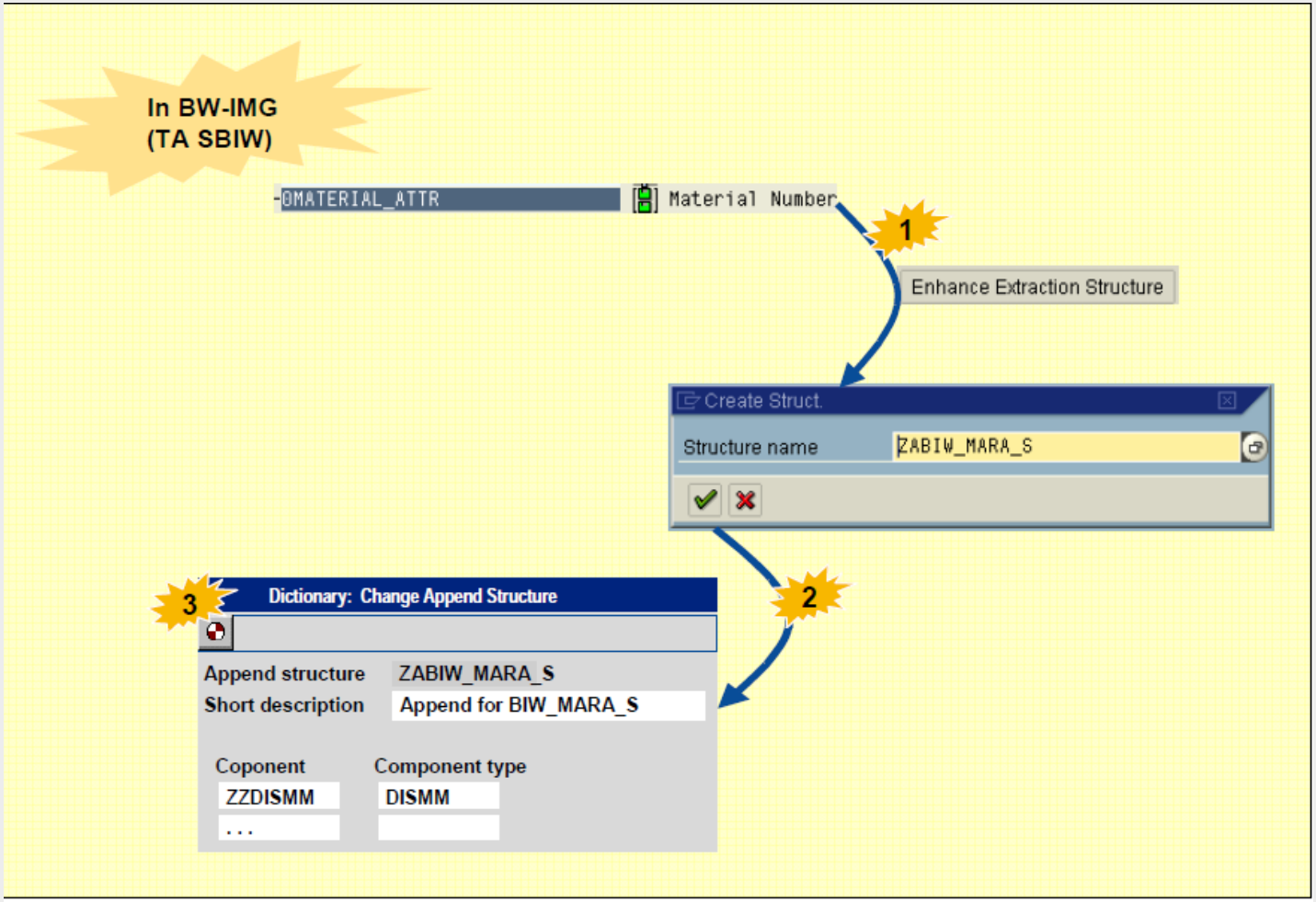
➢ **Reasons:**

- You want to add extra attributes to the characteristics (master data) shipped with the system
- You want to populate additional fields that you appended to the extract structure with data at the time of extraction
- You want to change/enhance texts or hierarchies

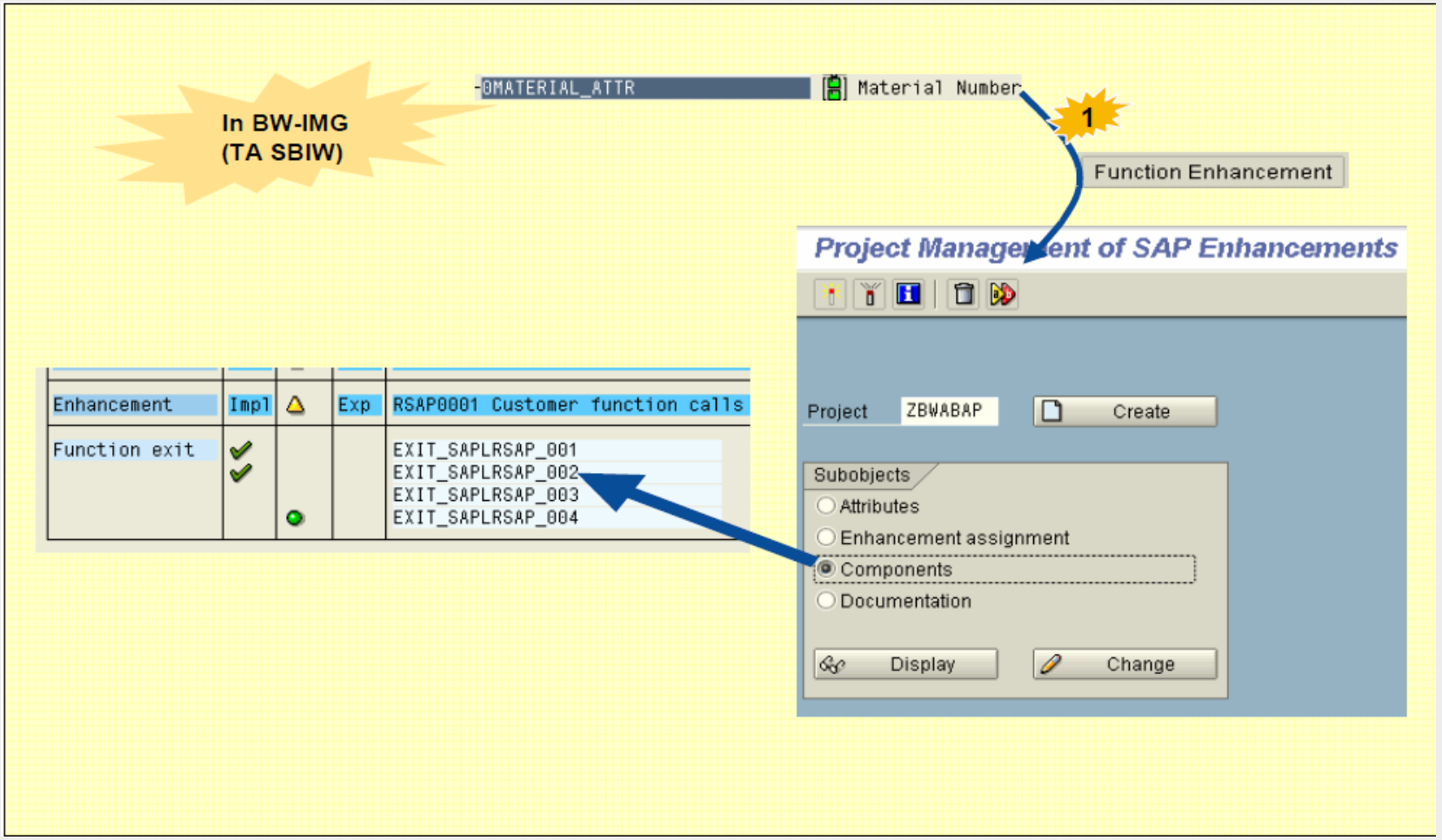
➢ **Prerequisites:**

- The additional data is available at the time the extraction takes place
- All information that is needed to determine the additional data clearly is available

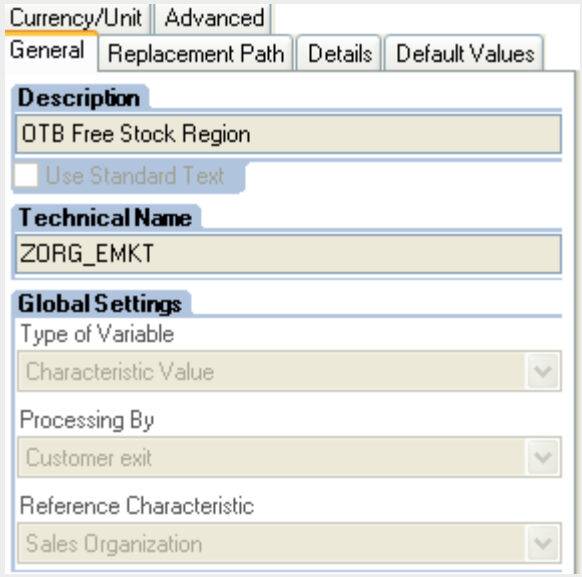
Creating and Maintaining Customer Appends



Developing Service API Function Enhancements

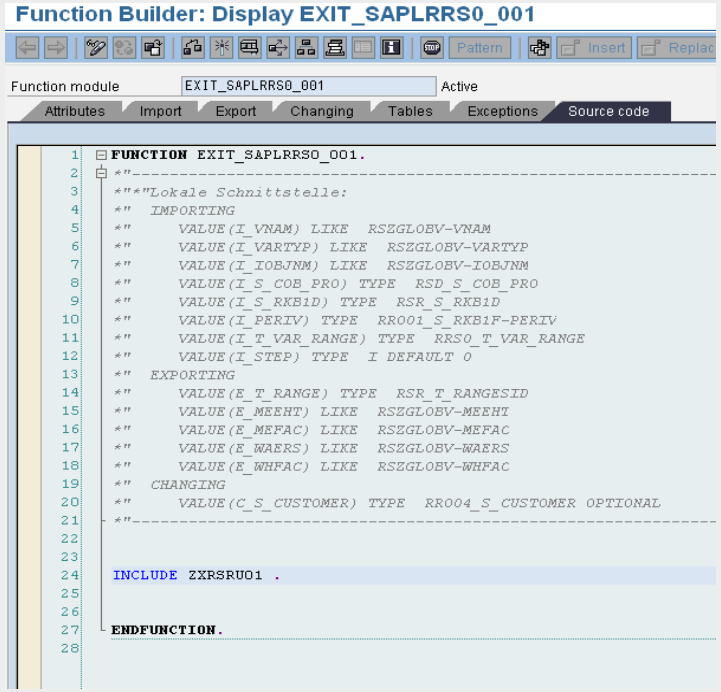


3. Query level Bex user Exits



Bex Variable

Exit name	Short text
☐ RSAP0001	Customer function calls in the service API
☑ RSR00001	BW: Enhancements for global variables in reporting
☐ RSR00002	BW: Virtual characteristics and key figures in Reporting
☐ RSR00003	BW: Moving characteristic values



Customer Exit in CMOD

BEx-Variables

- One way to fill a variable value is to use ABAP coding.
- We can differentiate between two alternatives:

1. Fill the value independently with a program
2. Fill the value dependent on an other variable value

- Default information:

- For filling variables the standard SAP Enhancement RSR00001 is used
- See the documentation related to this enhancement for details and example coding (transaction SMOD)
- In the function module of this enhancement you will find include ZXRSRU01 where you implement your program code
- The exit is called four times, called "steps" (I_STEP)

BEx-Variables (cont.)

- Default information:

- Step 1 (I_STEP = 1) is called before the processing of the variable pop-up and gets called for every variable of the processing type, "customer exit." You can use this step to fill your variable with a default or proposal value.

- Step 2 (I_STEP = 2) is called after processing of the variable pop-up. This step is called only for those variables that are not marked as "ready for input" and are set to "mandatory variable entry".

- Step 3 (I_STEP = 3) is called after all variable processing and gets called only once and not per variable. Here you can validate the user entries.

- Step 0 (I_STEP = 0) is called for variables that are used in authorizations objects.

4. Data Staging

- Overview
- InfoPackages
- DTPs
- Transformations
- Update rules

Sequential Order to Work with SAP BI Structures

The BW ABAP routines are processed sequentially for all records and all fields:

InfoPackage – Create dynamic filenames (for flat file import)

InfoPackage – Dynamic selection – selection criteria 1 ... n

Transfer rule – Start routine

Transfer rule – Dynamic routines to combine field by field from the transfer structure to the communication structure

Update rule – Start routine

Update rule – Dynamic routines to combine field by field from the communication struture to the infoprovider key figures or to the ODS-object data fields

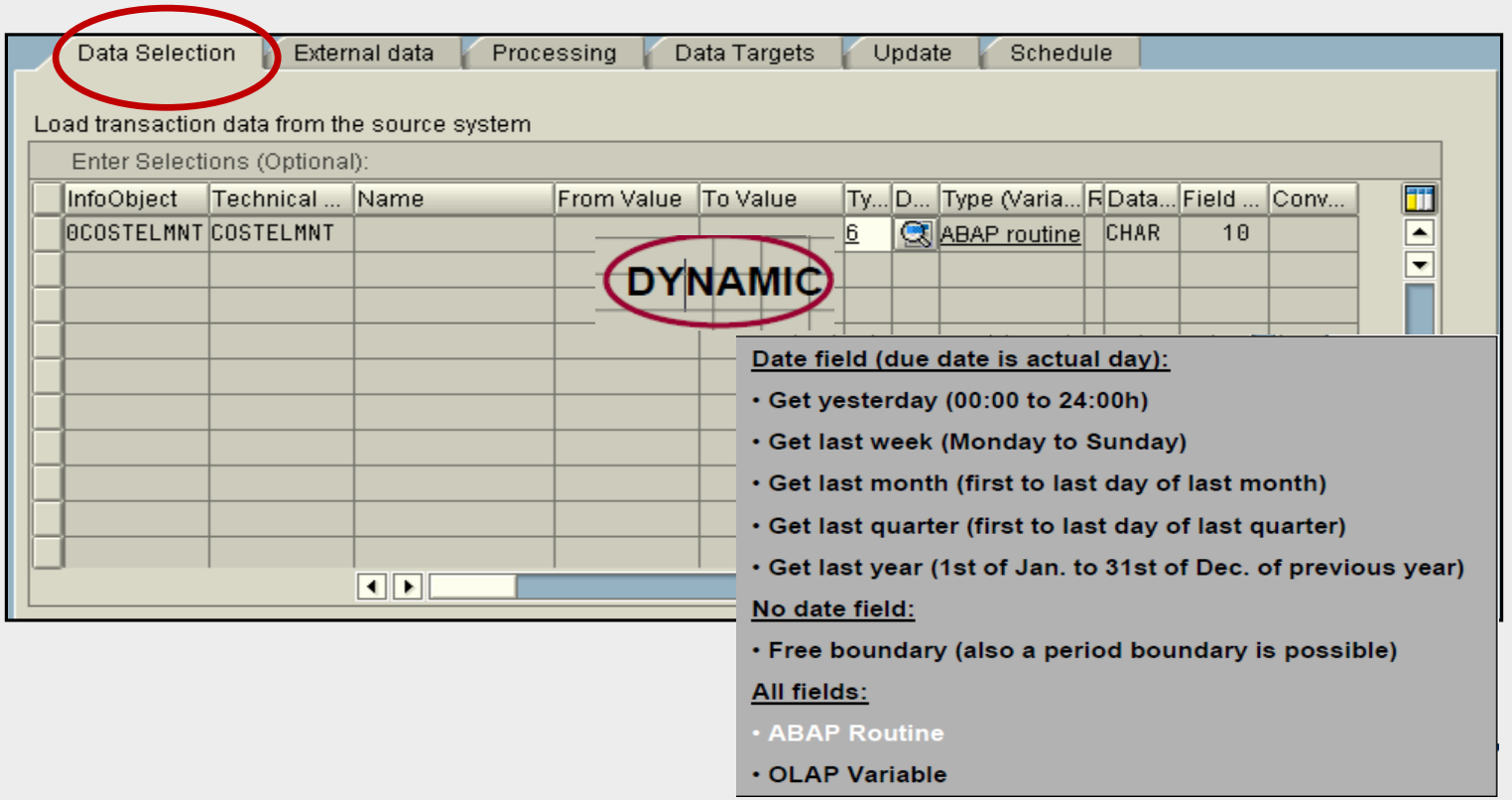
Update rule – Dynamic routines to combine field by field from the communication structure to the infoprovider characteristics or the ODS-object key fields

InfoPackage – Deletion or taking out of requests after the loading process:

Transformations (BI 7) – Start Routine, End Routine , Expert Routine

Info Packages : Selection Possibilities

➢InfoPackages provide static and dynamic selection criteria for every selection field for various characteristic types.

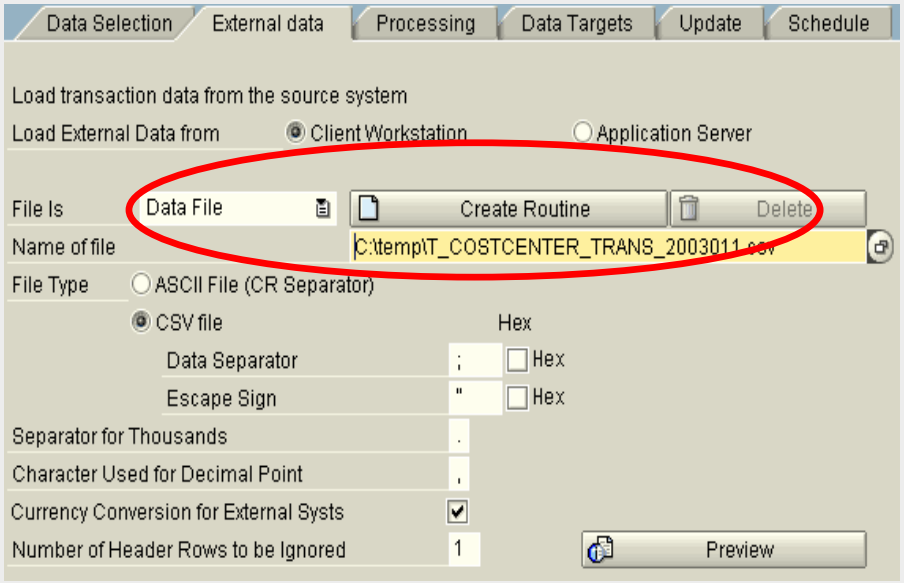


Infopackages : Dynamic Determination of Filenames

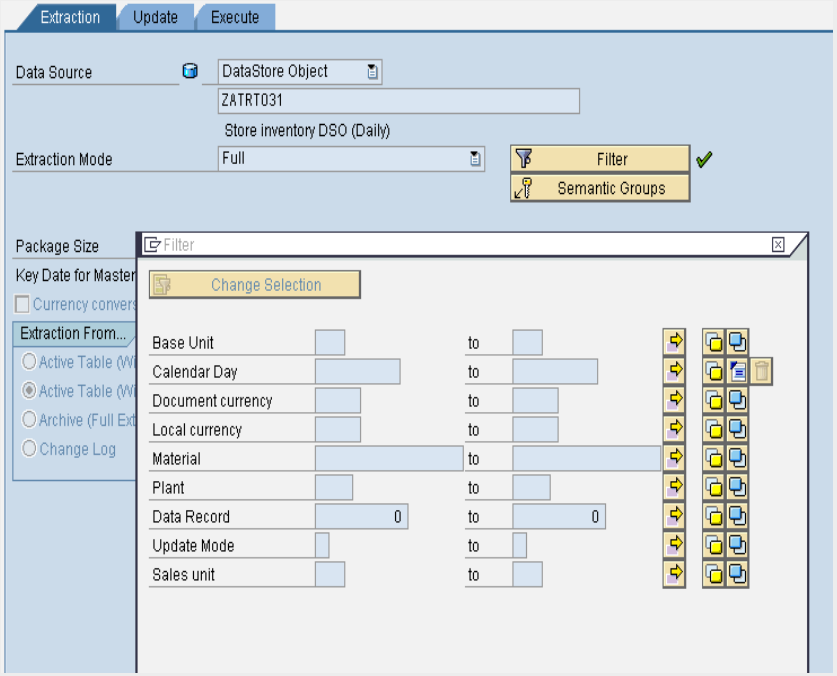
➢Should you upload from client workstation or the application server?

➢It is preferable to use the server path ☐ advantages:

- Data load in batch mode is possible
- Path is physically equal to all administrators
- Scenario in which to use it:
- Transaction data is delivered monthly by flat file
- The naming convention includes the name of the month within the filename
- Directory paths are maintained for every year
- Coding:
- Concatenate:
- Directory path
- Actual year
- Actual month



DTPs

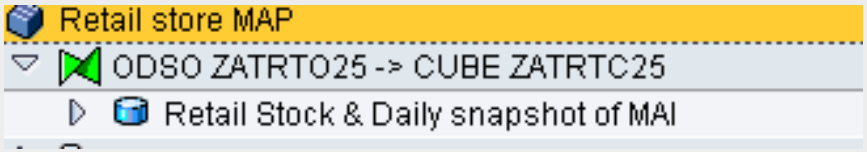


DTP Filter Selection Screen

```
form compute_CALDAY
  tables l_t_range structure rssdlrange
  using i_r_request type ref to IF_RSBK_REQUEST_ADMINTAB_VIEW
        i_fieldnm type RSFIELDNM
  changing p_subrc like sy-subrc.
*       Insert source code to current selection field
*$*$ begin of routine - insert your code only below this line        *-*
DATA : gv_calday_p TYPE cdobjectv,
       gv_calday   TYPE sy-datum.

  CLEAR : gv_calday_p,
          gv_calday.

  SELECT SINGLE value
    FROM zbw_upd_param
    INTO  gv_calday_p
    WHERE odsobject = 'SNAPSHOT'
          AND  fldname =  'CALDAY'.

  IF sy-subrc = 0.
    IF NOT gv_calday_p IS INITIAL.
*       gv_calday = gv_calday_p.
        gv_calday = gv_calday - 1.
    ELSE.
      gv_calday = sy-datum - 1.
    ENDIF.
  ENDIF.

  l_t_range-fieldname = i_fieldnm.
  l_t_range-sign = 'I' .
  l_t_range-option = 'EQ'.
  l_t_range-low = gv_calday.
  APPEND l_t_range.

  p_subrc = 0.
```

DTP Filter : ABAP Code

Transformations :



Retail Stock & Daily snapshot of MAP (ZATRTO25)

Posi	Ke	InfoObject	Ico	Descript.
1		0MATERIAL		Material.
2		0PLANT		Plant
3		0RECTOTSTCK		Receipt Quantity Total Stock
4		0ISSTOTSTCK		Issue Quantity Total Stock
5		0BASE_UOM		Base Unit of Measure
6		0RECVS_VAL		Value Received into Valuated Stock
7		0ISSVS_VAL		Value issued from valuated stock
8		ZSTOCKQTY		Stock Quantity
9		0PRICE_AVG		Moving Average Price / Periodic Unit Price
10		0CALDAY		Calendar Day
11		0LOC_CURRCY		Local currency
12		0CURRENCY		Currency Key
13		0RECORDMODE		BW Delta Process: Update Mode
14		ZSTKVALUE		Stock Value @ MAP

Rule Group: Standard Group

Ru	Rule Name	Posi	Ke	InfoObject	Ico	Descript.	Int
=	0MATERIAL	1		0MATERIAL		Material.	
	ZAFS_MAT	2		ZAFS_MAT		AFS Material	
=	0PLANT	3		0PLANT		Plant	
	0AF_STCAT	4		0AF_STCAT		AFS Stock Category	
=	0RECTOTSTCK	5		0RECTOTSTCK		Receipt Quantity Total Stock	
=	0ISSTOTSTCK	6		0ISSTOTSTCK		Issue Quantity Total Stock	
=	0RECVS_VAL	7		0RECVS_VAL		Value Received into Valuated Stock	
=	0ISSVS_VAL	8		0ISSVS_VAL		Value issued from valuated stock	
=	ZSTOCKQTY	9		ZSTOCKQTY		Stock Quantity	
=	0PRICE_AVG	10		0PRICE_AVG		Moving Average Price / Periodic Unit Price	
=	ZSTKVALUE	11		ZSTKVALUE		Stock Value @ MAP	
	ZLICSPAN	15		ZLICSPAN		Licensee & Spanish indicator	

Transformations : Start / End Routines

Transformation Display

Pattern | Pretty Printer | Routines Info.

```
83
84      TYPES: BEGIN OF gty_material,
85               material TYPE /bi0/oimaterial,
86               af_stcat TYPE /bi0/oiaf_stcat,
87               zafs_mat TYPE /bic/oizafs_mat,
88             END OF gty_material.
89
90      DATA: git_material TYPE STANDARD TABLE OF gty_material.
91
92  **--End of Changes 0MATERIAL V 1.0 (GKRISHNA)
93
94  *$*$ end of global - insert your declaration only before this line   *-*
95
96      METHODS
97        start_routine
98          IMPORTING
99            request                  type rsrequest
100           datapackid               type rsdatapid
101         EXPORTING
102           monitor                  type rstr_ty_t_monitors
103         CHANGING
104           SOURCE_PACKAGE              type _ty_t_SC_1
105         RAISING
106           cx_rsrout_abort.
107     METHODS
108       inverse_start_routine
109         IMPORTING
110           i_th_fields_outbound         TYPE rstran_t_field_inv
111           i_r_selset_outbound          TYPE REF TO cl_rsmds_set
112           i_is_main_selection          TYPE rs_bool
113           i_r_selset_outbound_complete TYPE REF TO cl_rsmds_set
114           i_r_universe_inbound         TYPE REF TO cl_rsmds_universe
115         CHANGING
116           c_th_fields_inbound          TYPE rstran_t_field_inv
117           c_r_selset_inbound           TYPE REF TO cl_rsmds_set
118           c_exact                      TYPE rs_bool.
119 ENDCLASS.                    "routine DEFINITION
```

Pattern | Pretty Printer | Routines Info.

```
260 *   <-> result package
261 *---------------------------------------------------------------------
262   METHOD end_routine.
263 *=== Segments ===
264
265     FIELD-SYMBOLS:
266       <RESULT_FIELDS>    TYPE _ty_s_TG_1.
267
268     DATA:
269       MONITOR_REC     TYPE rstmonitor.
270
271 *$*$ begin of routine - insert your code only below this line
272 ... "insert your code here
273 *--  fill table "MONITOR" with values of structure "MONITOR_REC"
274 *-   to make monitor entries
275 ... "to cancel the update process
276 *    raise exception type CX_RSROUT_ABORT.
277
278 FIELD-SYMBOLS: <fs> type st_zsd_d10,
279               <RP> TYPE _ty_s_TG_1.
280 If RESULT_PACKAGE is not INITIAL.
281   select doc_number req_date
282   from /bic/aZSD_D1000
283   into table t_zsd_d10
284   FOR ALL ENTRIES IN RESULT_PACKAGE
285   where doc number  = RESULT_PACKAGE-doc number.
```

Start Routine : Classic usage

➢Particularly the start routine of the transformation gives.

- "classic" scenarios of usage:
- Selective deletion of records delivered with the data package
- Coding example:

- DELETE SOURCE_PACKAGE WHERE VKORG = "1000."

➢Advantage:

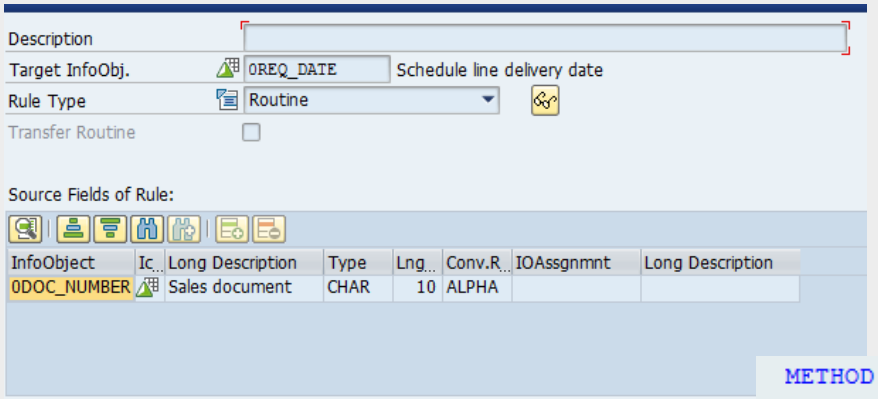
- Smaller data package -- loading performance improves
- Reasons why (examples):
- The InfoPackage from the data source does not offer selection criteria
- The processing time with a selection from the InfoPackage sent to the source system takes considerably more time than without selection
- The deletion criteria is easy to code within the start routine
- Fill an internal table from a DDIC select
- Complex data cleansing or data consolidation from various
- source systems eventually with a look up as ETL process

End Routine : Classic usage

- Particularly the End routine of the transformation gives “classic” scenarios of usage:
- Complex logic for fields which need look up to another DSO or master data to get data for those fields which are not present at Source Targets
- “Result_package” is internal table which contain the data

Field Routine: Classic usage

➢"Source_fields" contains the row level record data at transformation and in order to update value of field we have update "RESULT". We can use field routine for small if. Else logic or simple operation like get Net value from Price and Order qty.



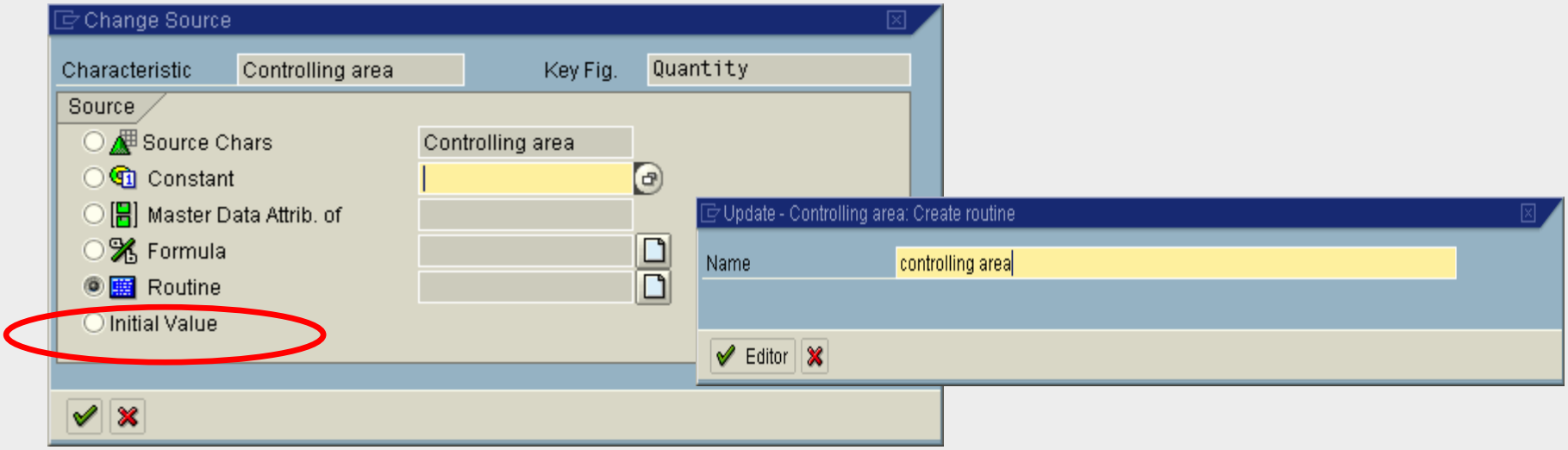

```
METHOD compute_0REQ_DATE.

*   IMPORTING
*     request      type rsrequest
*     datapackid   type rsdatapid
*     SOURCE_FIELDS-DOC_NUMBER TYPE /BI0/OIDOC_NUMBER
*   EXPORTING
*     RESULT type _ty_s_TG_1-REQ_DATE

  DATA:
    MONITOR_REC    TYPE rsmonitor.
```

Update Rules : InfoCube Characteristic ODS-Object Key Field

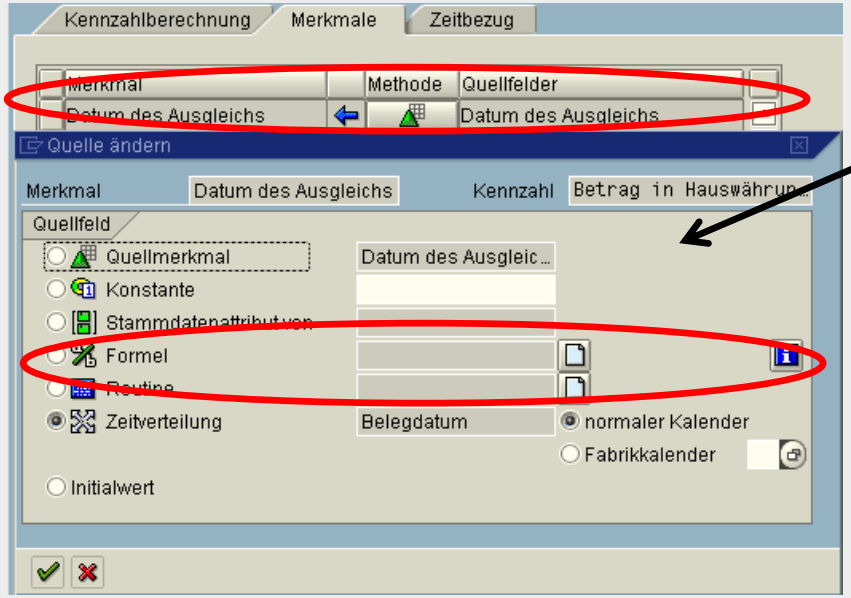
➢**Interface:**

- Like in the start routine global declaration within the header.
- The form routine “compute_key_field” delivers the following parameters:
- The table MONITOR for monitoring
- The communication structure COMM_STRUCTURE with all fields from the data source
- The current record number of the loop over the data package
- The number of total records
- RESULT delivers to the marked characteristic the result value
- RETURNCODE says whether the current record will be processed or not
- ABORT clarifies whether the whole data package will be processed or not

Update Rules : InfoCube Characteristic ODS-Object Key Field



Update Rules : Time References



Within update rules there are various possibilities to reference time dependencies:

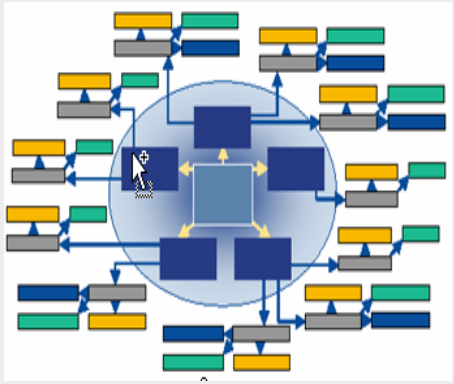
<u>Characteristic update rule with time reference</u>

- **It is possible to update data target characteristics of type "DATE" via an automatic time distribution**
- **The field is fed from a data field out of the communication structure**
- **For example, the data target characteristic calendar day is fed from the source characteristic calendar month**
- **The effect is that every summarized key figure has to be distributed to the calendar days of the month**
- **In addition to that, it can be defined whether the company calendar has to be used**

<u>Time characteristics</u>

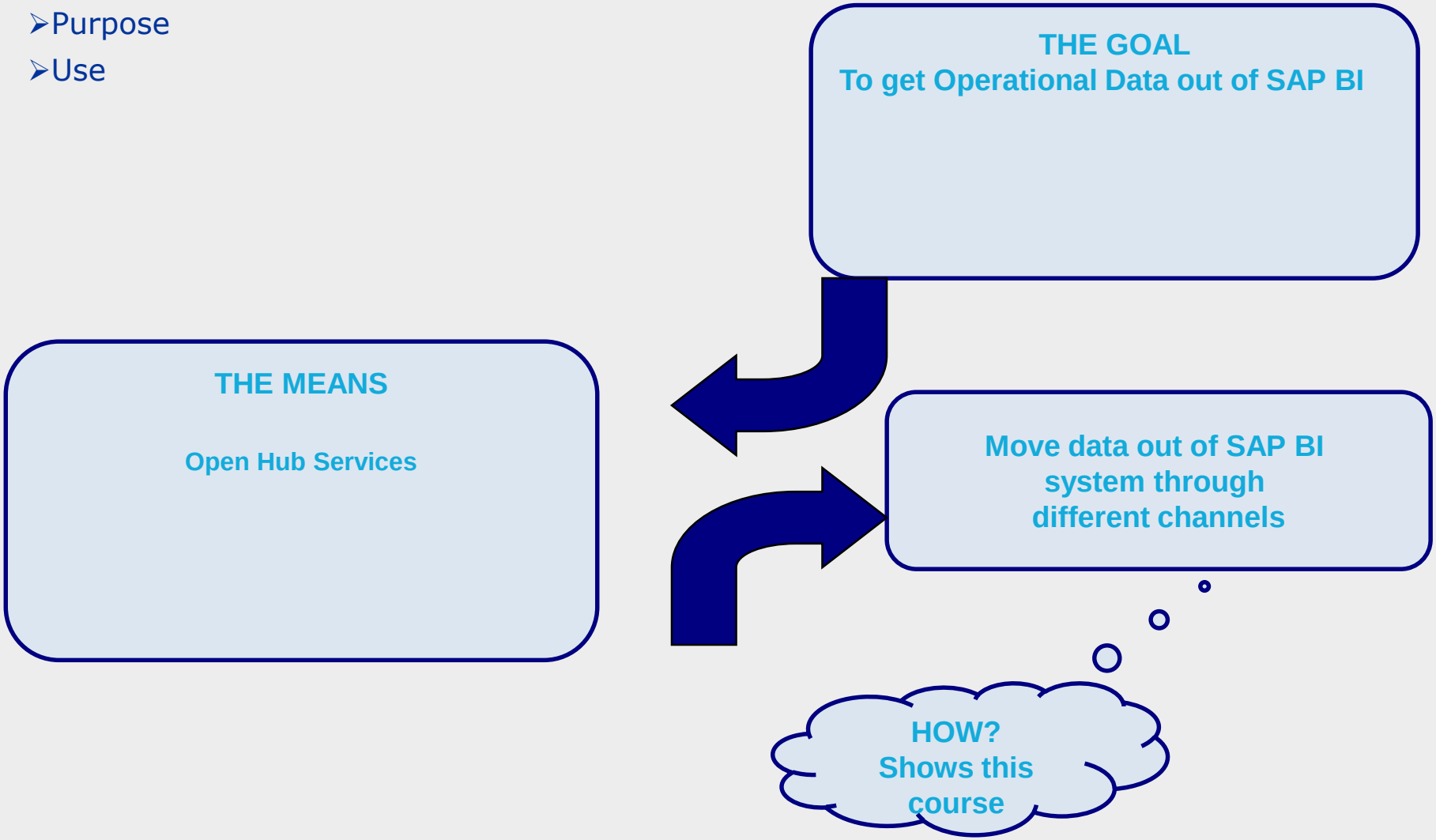
- **Within the frame time reference various data target time characteristics can be filled automatically from only one time characteristic of the data source**

5. Open Hub Destination

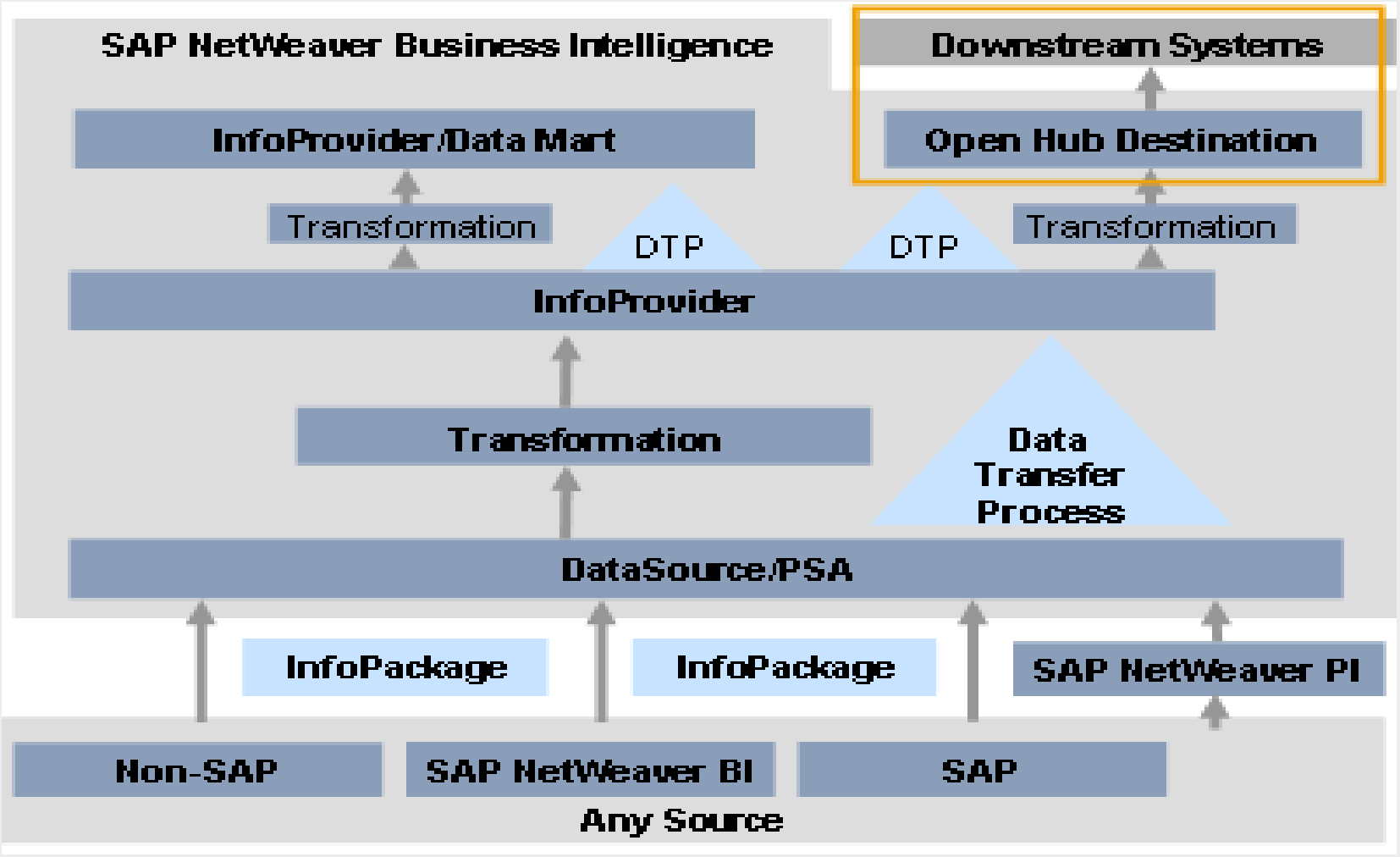


Prepare Me

- Purpose
- Use



OHS


SAP NetWeaver Business Intelligence
Downstream Systems
InfoProvider/Data Mart
Open Hub Destination
Transformation
DTP
DTP
Transformation
InfoProvider
Transformation
Data Transfer Process
DataSource/PSA
InfoPackage
InfoPackage
SAP NetWeaver PI
Non-SAP
SAP NetWeaver BI
SAP
Any Source

Purpose

- 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 ensures controlled distribution across multiple systems.

- Open hub destination defines the target to which the data is transferred.

Use

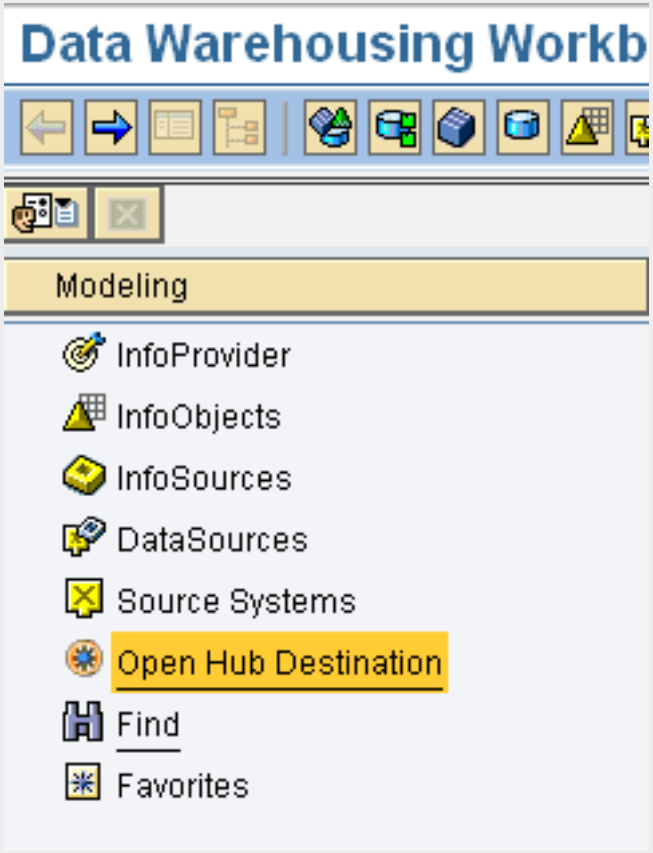
➢Database tables (in the database for the BI system) and flat files can act as open hub destinations. You can extract the data from a database to non-SAP systems using APIs and a third-party tool.

➢In earlier releases, the open hub destination was called as InfoSpoke and it was not as tightly integrated into the data flow as Open hub destination is.

OHS – Types

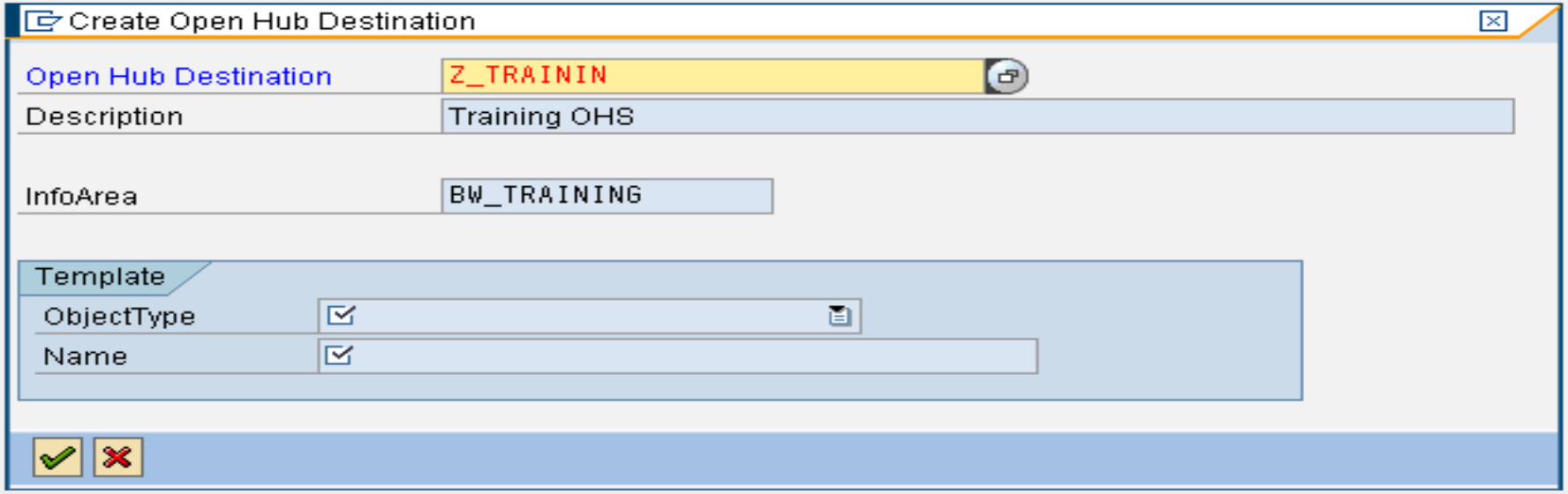
- Using OHS we can write data from SAP BI systems into below mentioned data targets:
  - Database Tables.
  - Flat File.
  - Third-Party Tool.
- The open hub destination contains all the information about a data target:
  - Type of destination.
  - Name of the flat file or database table and its properties.
  - Field list and its properties.

OHS – Access

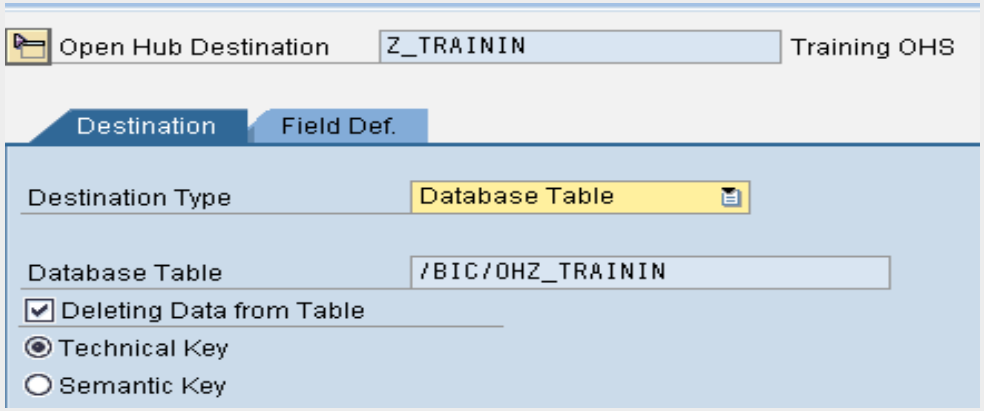


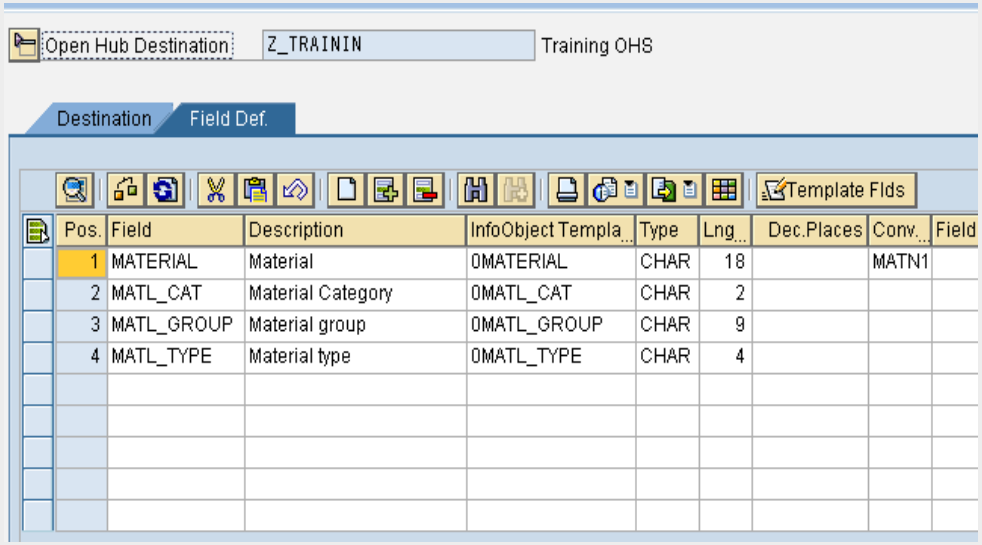
Using RSA1 transaction, Data Warehousing Workbench, we can access OHS under the Modeling tab

OHS - Creation



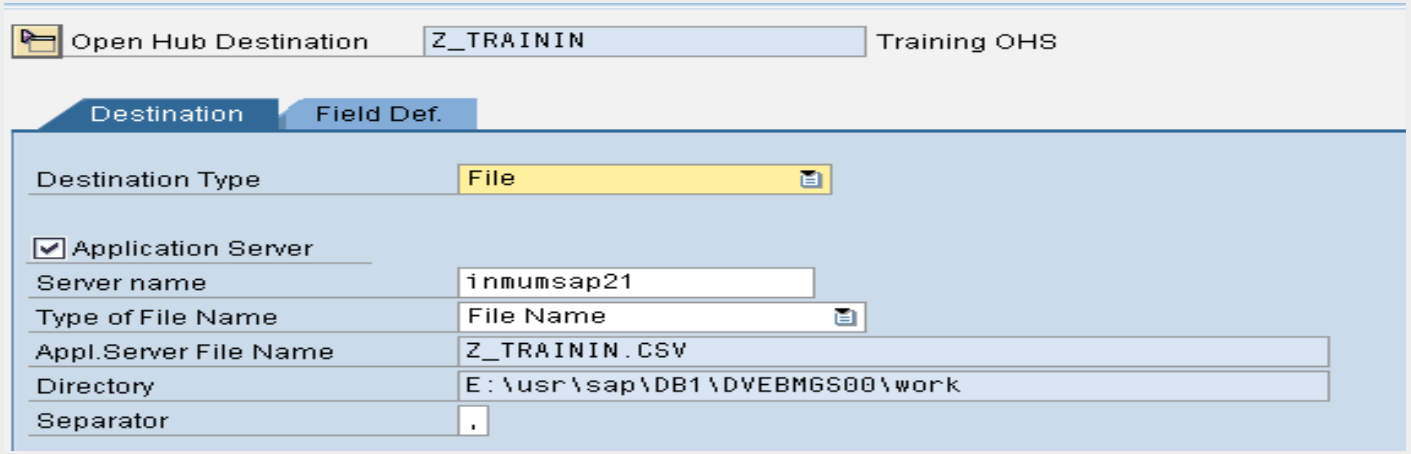
OHS – using Database table

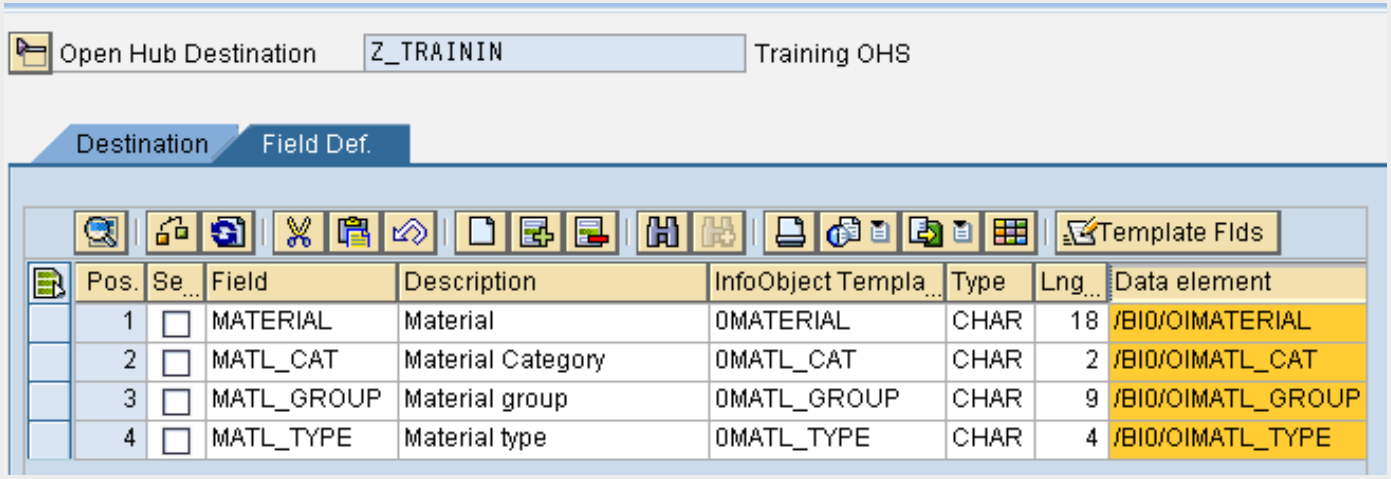




Pos.	Field	Description	InfoObject Templa...	Type	Lng...	Dec.Places	Conv...	Field
1	MATERIAL	Material	0MATERIAL	CHAR	18		MATN1	
2	MATL_CAT	Material Category	0MATL_CAT	CHAR	2			
3	MATL_GROUP	Material group	0MATL_GROUP	CHAR	9			
4	MATL_TYPE	Material type	0MATL_TYPE	CHAR	4			

OHS – using Flat File



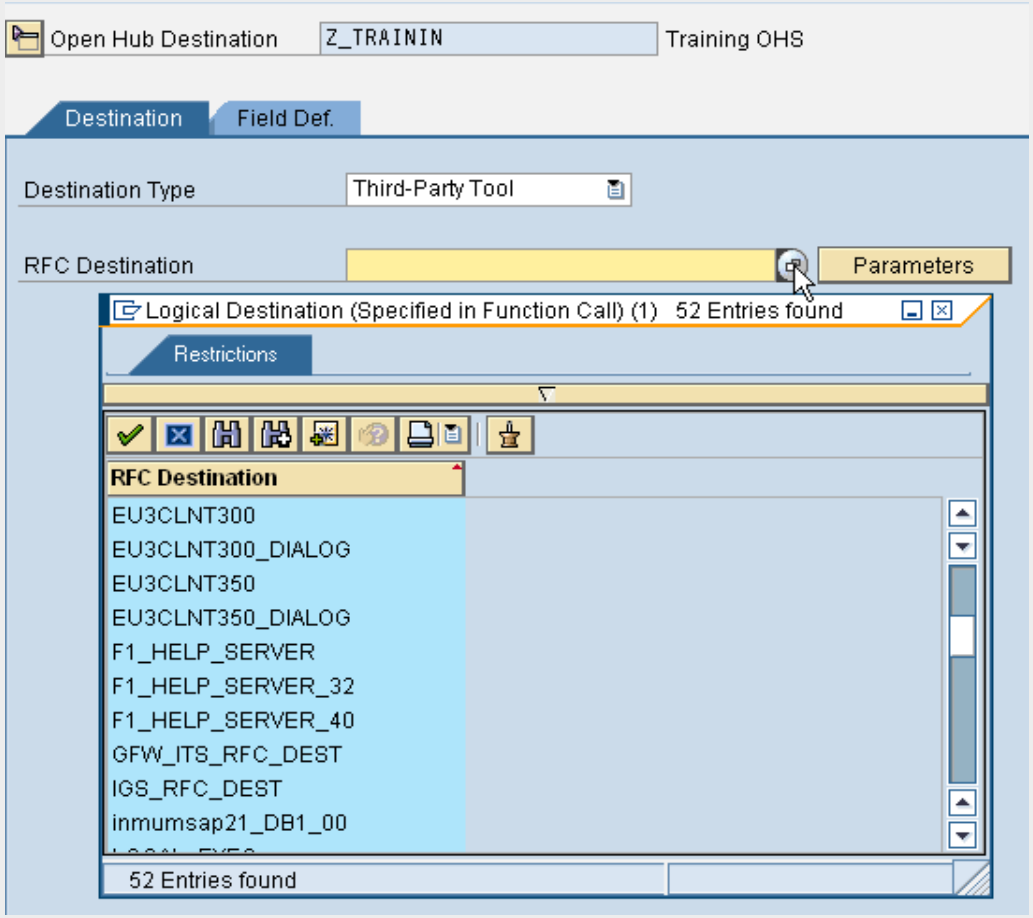


Pos.	Se...	Field	Description	InfoObject Templa...	Type	Lng...	Data element
1	☐	MATERIAL	Material	0MATERIAL	CHAR	18	/BI0/OIMATERIAL
2	☐	MATL_CAT	Material Category	0MATL_CAT	CHAR	2	/BI0/OIMATL_CAT
3	☐	MATL_GROUP	Material group	0MATL_GROUP	CHAR	9	/BI0/OIMATL_GROUP
4	☐	MATL_TYPE	Material type	0MATL_TYPE	CHAR	4	/BI0/OIMATL_TYPE

OHS – using Third-Party Tool

➢ Using Third-party tool, we can transfer data from SAP BI system, to different systems by linking them to BI system.

➢ You create an RFC destination for your third-party tool and enter it in the definition of the OHS.



Transactions Used

- RSA1 : Data Warehousing Workbench.

HelpMe

Browse through the below links for Self Study

1. http://help.sap.com/saphelp_nw70ehp2/helpdata/en/e3/e60138fede083de10000009b38f8cf/frameset.htm

2. http://service.sap.com/bi