Usage of ABAP in BI

Applies to:

SAP BW 3.5, SAP BI 7.0 etc., For more information, visit the EDW homepage.

Summary

This paper has been prepared to give an insight view about the usage of ABAP in BI by illustrating specific examples.

Author: Poornima Gayatri

Company: Deloitte Consulting

Created on: 17 May 2011

Author Bio

Poornima Gayatri Chennur is currently working in Deloitte Consulting India Pvt. Ltd. She is working on SAP BW/BI from last 3.10 years.

Table of Contents

Introduction

In some cases we need to enhance functionalities of BI by using ABAP technology to manipulate and transform the data according to the user's requirement. We can use ABAP coding majorly in

- InfoPackage Routines
- Filters in DTP
- Start Routine
- End Routine
- Expert Routine
- Field Routine
- Customer Exit for Query Variables

InfoPackage Routines

For InfoPackages Routines can be created at:

1) In Extraction Tab – A routine can be created that determines the name of your file.
2) In Data Selection Tab – Routines can be created at Data Selection tab page to determine the data selection from source systems.

In Extraction Tab

A routine can be created at Extraction page to determine the name of the file. The data can be loaded either from presentation server or from application server.

Use of the Routine

If the user changes the flat file which contains data to be loaded time to time, then the file has to be updated manually every time it is being changed. Instead a routine can be created to load the file.

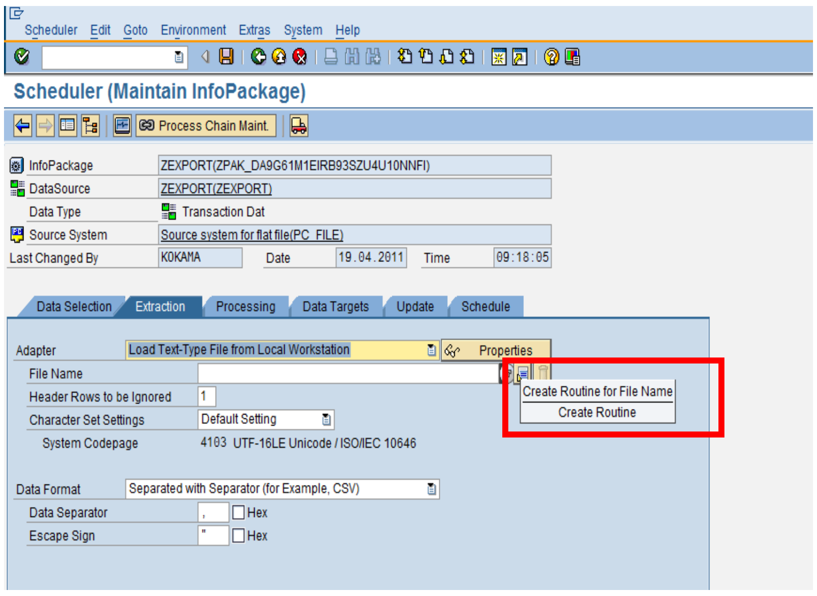
Whenever the InfoPackage runs, the routine will be executed and according to the logic, the data will be selected.

Example

In this example, a scenario is taken, in which the file will be selected using a routine and the data from the file is loaded according to the routine.

The file is selected from the presentation server. Whenever the InfoPackage is executed, it will check for the file with .CSV extension and loads the file.

In the extraction tab, click on icon 'Create Routine'.

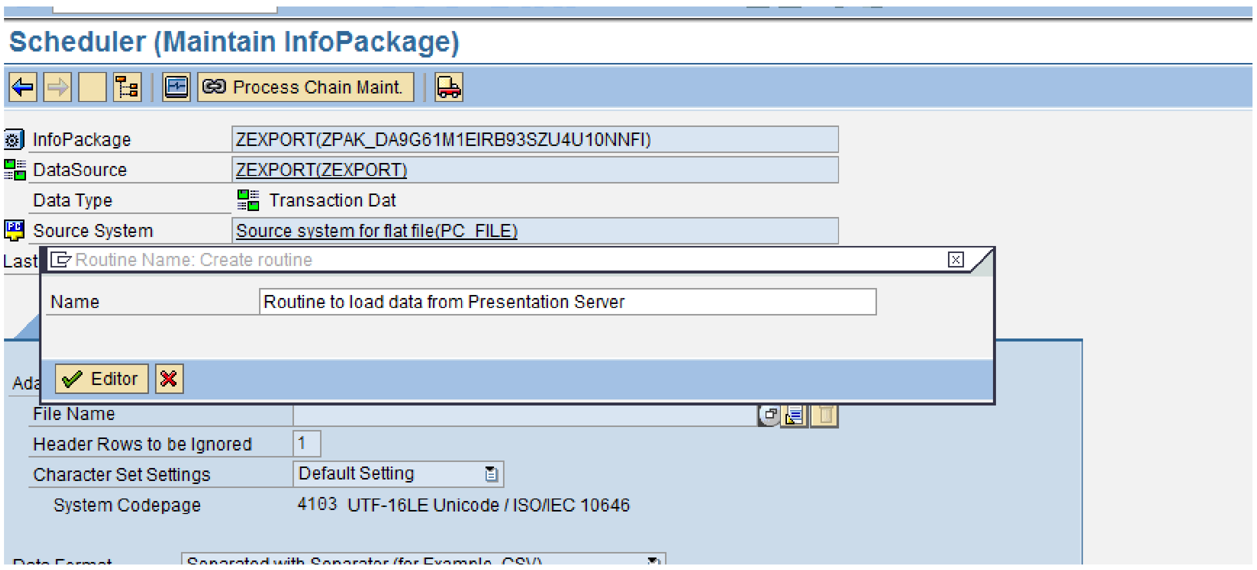


Scheduler (Maintain InfoPackage)

InfoPackage	ZEXPORT(ZPAK_DA9G61M1EIRB93SZU4U10NNFI)
DataSource	ZEXPORT(ZEXPORT)
Data Type	Transaction Dat
Source System	Source system for flat file(PC_FILE)
Last Changed By	KOKAMA Date 19.04.2011 Time 09:18:05

Adapter	Load Text-Type File from Local Workstation
File Name	
Header Rows to be Ignored	1
Character Set Settings	Default Setting
System Codepage	4103 UTF-16LE Unicode / ISO/IEC 10646
Data Format	Separated with Separator (for Example, CSV)
Data Separator	, Hex
Escape Sign	" Hex

A pop up appears to enter the name of the routine. Enter the Routine name and click on Editor.


Scheduler (Maintain InfoPackage)
Process Chain Maint.
Routine Name: Create routine
Editor

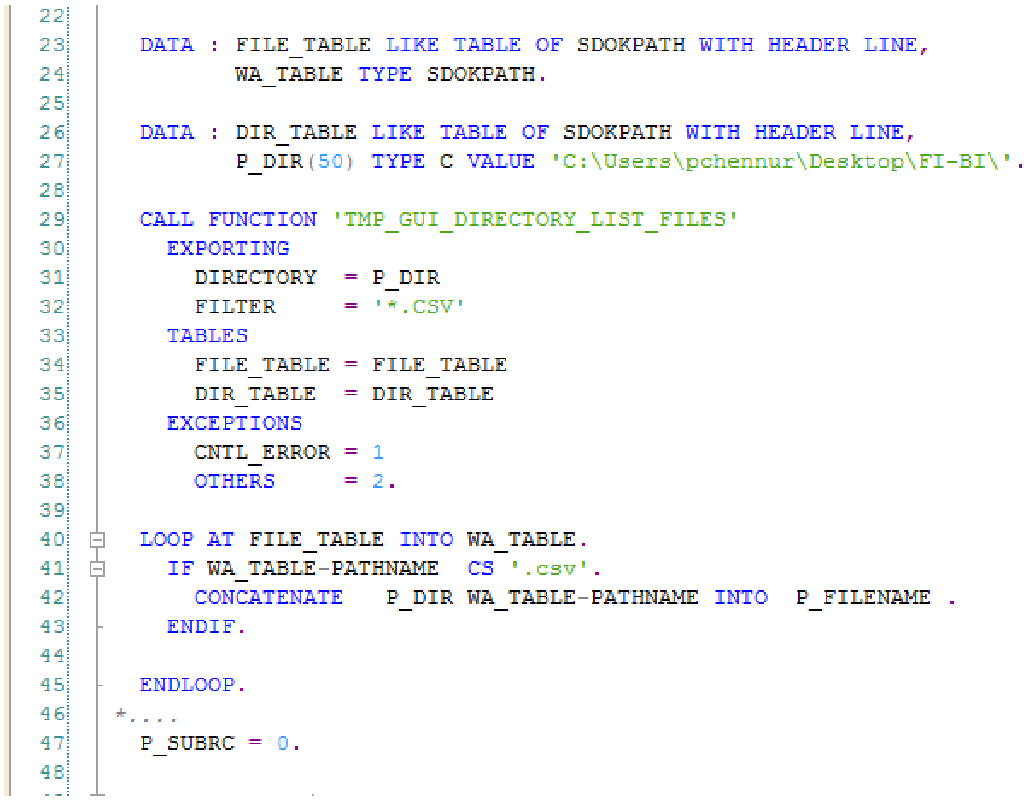
InfoPackage	ZEXPORT(ZPAK_DA9G61M1EIRB93SZU4U10NNFI)
DataSource	ZEXPORT(ZEXPORT)
Data Type	Transaction Dat
Source System	Source system for flat file(PC_FILE)

Name	Routine to load data from Presentation Server

File Name	
Header Rows to be Ignored	1
Character Set Settings	Default Setting
System Codepage	4103 UTF-16LE Unicode / ISO/IEC 10646

Editor appears where the code can be written. In this example, the routine is written to check whether a .csv file appears in the folder and the data will be loaded from the file accordingly.

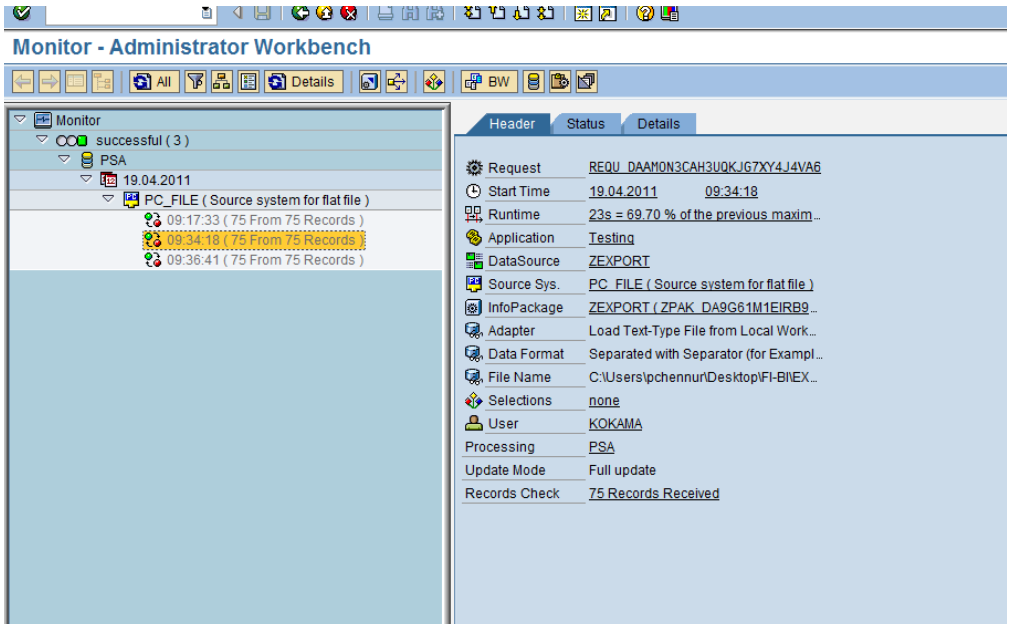
The code is given below.



```
DATA : FILE_TABLE LIKE TABLE OF SDOKPATH WITH HEADER LINE,
       WA_TABLE TYPE SDOKPATH.

DATA : DIR_TABLE LIKE TABLE OF SDOKPATH WITH HEADER LINE,
       P_DIR(50) TYPE C VALUE 'C:\Users\pchennur\Desktop\FI-BI\'.

CALL FUNCTION 'TMP_GUI_DIRECTORY_LIST_FILES'
  EXPORTING
    DIRECTORY  = P_DIR
    FILTER     = '*.CSV'
  TABLES
    FILE_TABLE = FILE_TABLE
    DIR_TABLE  = DIR_TABLE
  EXCEPTIONS
    CNTL_ERROR = 1
    OTHERS     = 2.

LOOP AT FILE_TABLE INTO WA_TABLE.
  IF WA_TABLE-PATHNAME  CS '.csv'.
    CONCATENATE   P_DIR WA_TABLE-PATHNAME INTO  P_FILENAME .
  ENDIF.

ENDLOOP.
*....
P_SUBRC = 0.
```

Save the routine.

Schedule the InfoPackage. The records will be loaded from the file.



Monitor
successful (3)
PSA
19.04.2011
PC_FILE (Source system for flat file)
09:17:33 (75 From 75 Records)
09:34:18 (75 From 75 Records)
09:36:41 (75 From 75 Records)

Request	REQU_DAAMON3CAH3UQKJG7XY4J4VA6
Start Time	19.04.2011 09:34:18
Runtime	23s = 69.70 % of the previous maxim…
Application	Testing
DataSource	ZEXPORT
Source Sys.	PC_FILE (Source system for flat file)
InfoPackage	ZEXPORT (ZPAK_DA9G61M1EIRB9…
Adapter	Load Text-Type File from Local Work…
Data Format	Separated with Separator (for Exampl…
File Name	C:\Users\pchennur\Desktop\FI-BI\EX…
Selections	none
User	KOKAMA
Processing	PSA
Update Mode	Full update
Records Check	75 Records Received

In Data Selection Tab

Use of InfoPackage Routine

To load data into BI from the source system periodically and to change the contents of the selection field each time of the load, it is possible to define the selections for the fields in the InfoPackage. The variable change can be implemented 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.

Features

We can use either ABAP Routine (Variable type 6) or OLAP Variables (Variable type 7) in the Data selection while loading the data using InfoPackages.

Selections using an ABAP routine (variable type 6)

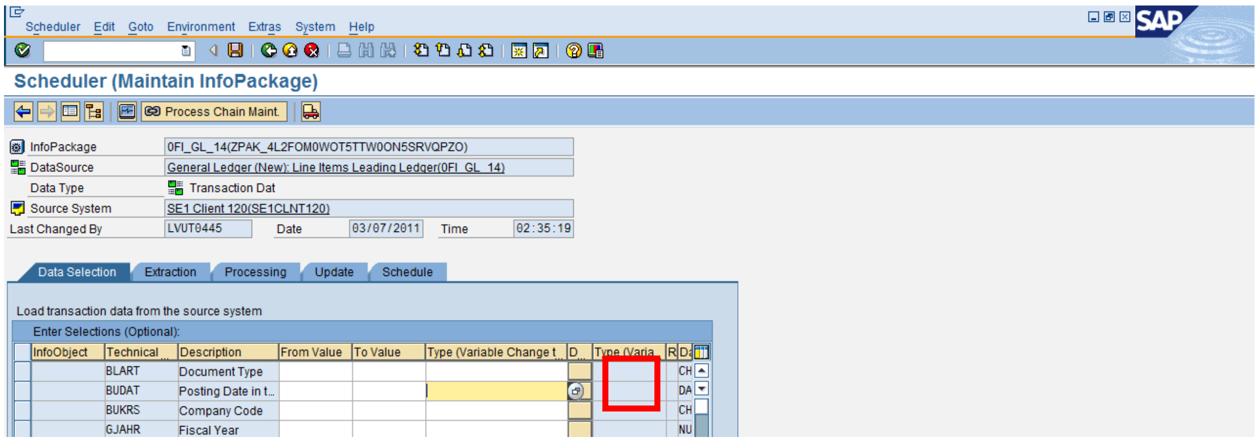
An ABAP program can be written to restrict the requested data of an InfoObject or field. Then, the select type should be given as ‘6’. A screen appears on which the name for the ABAP routine can be given. After entering the name, the editor appears and the code can be entered here. The ABAP routine has access to all selection fields and is the last to be processed at runtime.

Explicitly following definitions can be made 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.

Example

The below example is demonstrated by restricting the data for the filed 'BUDAT' in the InfoPackage created for the DataSource '0FI_GL_14'.In this example, the InfoPackage will pick up the data from the date which is less than the current date.

In the Data Selection Tab of the InfoPackage, select the Type as 'ABAP Routine'.


Scheduler Edit Goto Environment Extras System Help
Process Chain Maint.
Data Selection Extraction Processing Update Schedule

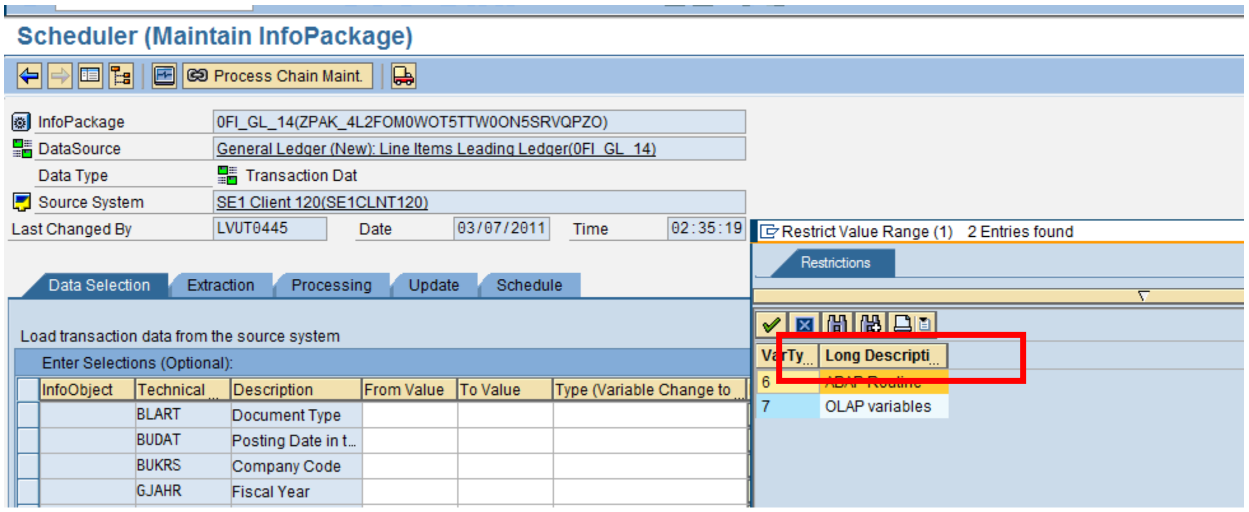
Scheduler (Maintain InfoPackage)

InfoPackage	0FI_GL_14(ZPAK_4L2FOM0WOT5TTW0ON5SRVQPZO)
DataSource	General Ledger (New): Line Items Leading Ledger(0FI_GL_14)
Data Type	Transaction Dat
Source System	SE1 Client 120(SE1CLNT120)
Last Changed By	LVUT0445 Date 03/07/2011 Time 02:35:19

Load transaction data from the source system

Enter Selections (Optional):

InfoObject	Technical ...	Description	From Value	To Value	Type (Variable Change t...	D...	Type (Varia...	R	Da
	BLART	Document Type							CH
	BUDAT	Posting Date in t...							DA
	BUKRS	Company Code							CH
	GJAHR	Fiscal Year							NU



Scheduler (Maintain InfoPackage)

InfoPackage	0FI_GL_14(ZPAK_4L2FOM0WOT5TTW0ON5SRVQPZO)
DataSource	General Ledger (New): Line Items Leading Ledger(0FI_GL_14)
Data Type	Transaction Dat
Source System	SE1 Client 120(SE1CLNT120)
Last Changed By	LVUT0445 Date 03/07/2011 Time 02:35:19

Load transaction data from the source system

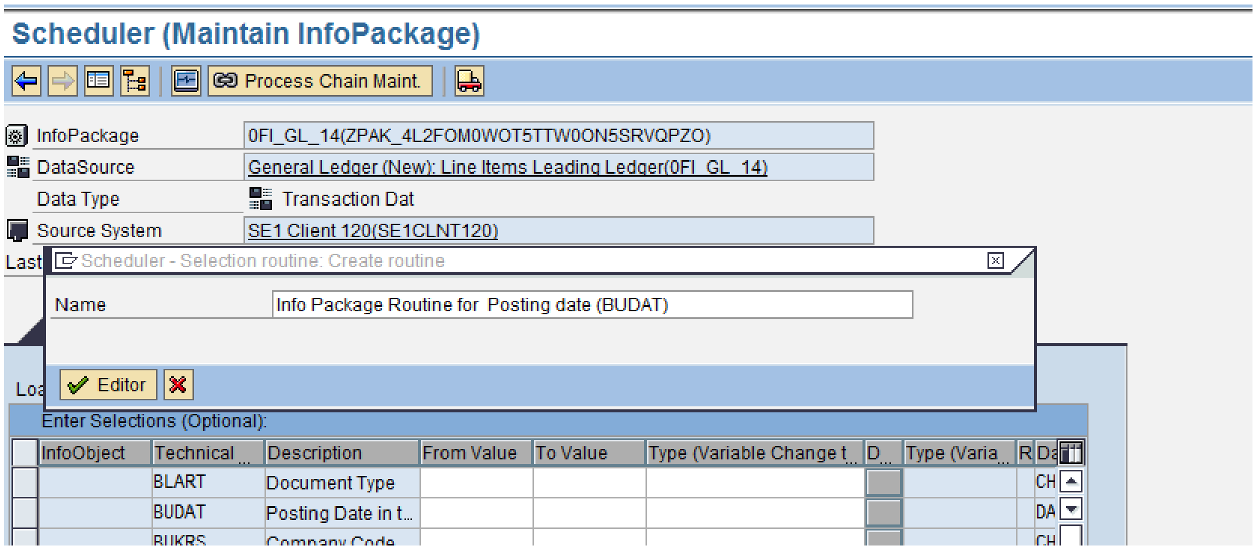
Enter Selections (Optional):

InfoObject	Technical ...	Description	From Value	To Value	Type (Variable Change to
	BLART	Document Type			
	BUDAT	Posting Date in t...			
	BUKRS	Company Code			
	GJAHR	Fiscal Year			

Restrict Value Range (1) 2 Entries found

VarTy...	Long Descripti...
6	ABAP Routine
7	OLAP variables

There will be an automatic pop up to enter the description for the routine.



Scheduler (Maintain InfoPackage)

InfoPackage	0FI_GL_14(ZPAK_4L2FOM0WOT5TTW0ON5SRVQPZO)
DataSource	General Ledger (New): Line Items Leading Ledger(0FI_GL_14)
Data Type	Transaction Dat
Source System	SE1 Client 120(SE1CLNT120)

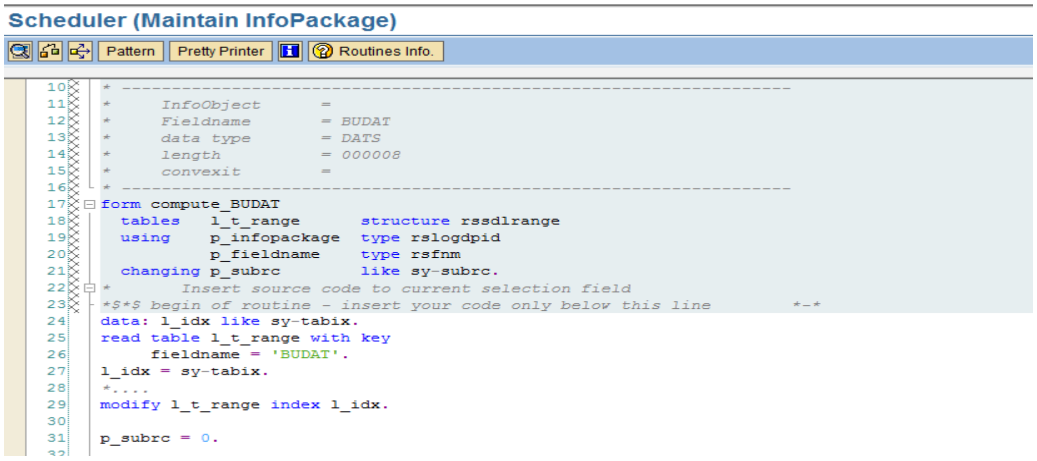
Scheduler - Selection routine: Create routine

Name: Info Package Routine for Posting date (BUDAT)

Enter Selections (Optional):

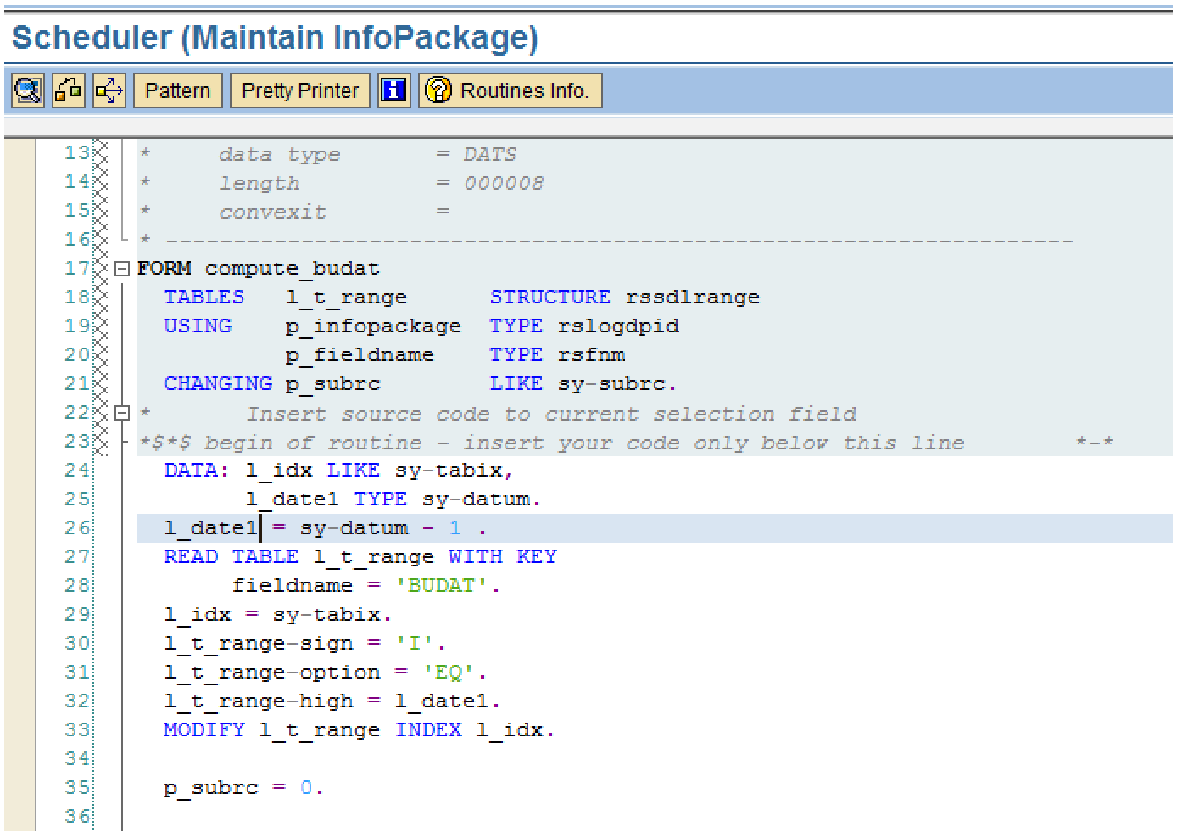
InfoObject	Technical ...	Description	From Value	To Value	Type (Variable Change t...	D...	Type (Varia...	R	Da
	BLART	Document Type							CH
	BUDAT	Posting Date in t...							DA
	BUKRS	Company Code							CH

Once it enters into ABAP editor, the code can be written accordingly.



```
*  -------------------------------------------------------------------
*      InfoObject        =
*      Fieldname         = BUDAT
*      data type         = DATS
*      length            = 000008
*      convexit          =
*  -------------------------------------------------------------------
form compute_BUDAT
  tables     l_t_range      structure rssdlrange
  using      p_infopackage  type rslogdpid
             p_fieldname    type rsfnm
  changing p_subrc          like sy-subrc.
*        Insert source code to current selection field
*$*$ begin of routine - insert your code only below this line        *-*
data: l_idx like sy-tabix.
read table l_t_range with key
     fieldname = 'BUDAT'.
l_idx = sy-tabix.
*....
modify l_t_range index l_idx.

p_subrc = 0.
```

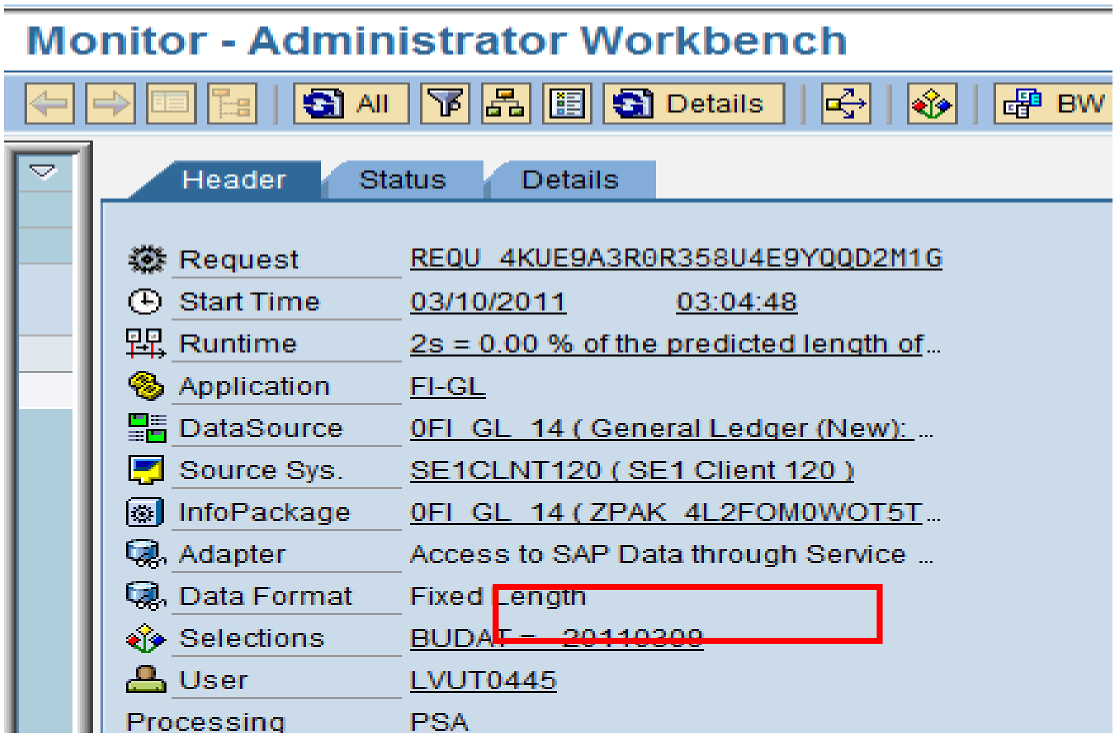
Logic for the code to enter the value of the Posting date which is less than current day.



```
*      data type         = DATS
*      length            = 000008
*      convexit          =
*  -------------------------------------------------------------------
FORM compute_budat
  TABLES     l_t_range      STRUCTURE rssdlrange
  USING      p_infopackage  TYPE rslogdpid
             p_fieldname    TYPE rsfnm
  CHANGING p_subrc          LIKE sy-subrc.
*        Insert source code to current selection field
*$*$ begin of routine - insert your code only below this line        *-*
  DATA: l_idx LIKE sy-tabix,
        l_date1 TYPE sy-datum.
  l_date1 = sy-datum - 1 .
  READ TABLE l_t_range WITH KEY
       fieldname = 'BUDAT'.
  l_idx = sy-tabix.
  l_t_range-sign = 'I'.
  l_t_range-option = 'EQ'.
  l_t_range-high = l_date1.
  MODIFY l_t_range INDEX l_idx.

  p_subrc = 0.
```

Monitor Screen with Selection Criteria

The InfoPackage is run on 10/03/2011. So we can see that the date displayed here is 09/03/2011.



Request	REQU_4KUE9A3R0R358U4E9YQQD2M1G
Start Time	03/10/2011 03:04:48
Runtime	2s = 0.00 % of the predicted length of...
Application	FI-GL
DataSource	0FI_GL_14 (General Ledger (New): ...
Source Sys.	SE1CLNT120 (SE1 Client 120)
InfoPackage	0FI_GL_14 (ZPAK_4L2FOM0WOT5T...
Adapter	Access to SAP Data through Service ...
Data Format	Fixed Length
Selections	BUDAT = 20110309
User	LVUT0445
Processing	PSA

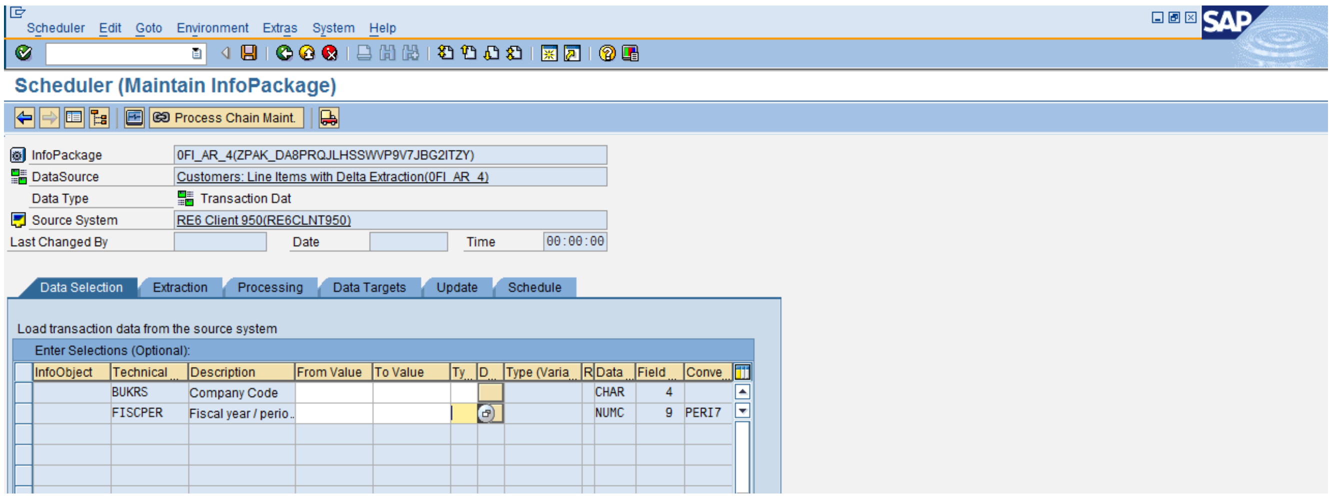
Selections using a variable (variable type 7)

(OLAP) variables are used as placeholders for values of InfoObjects. They are replaced with concrete values during a data request. Then, select type should be given as ‘7’. A screen appears on which you can select the variables.

Example

The below example is demonstrated by restricting the data using and OLAP Variable for the filed ‘FISCPER (Fiscal Period)’ in the InfoPackage created for the DataSource ‘0FI_AR_4’.In this example, the InfoPackage will pick up the data from the data only for current Fiscal Year.

In the Data Selection Tab of the InfoPackage, select the Type as **‘OLAP Variable’.**



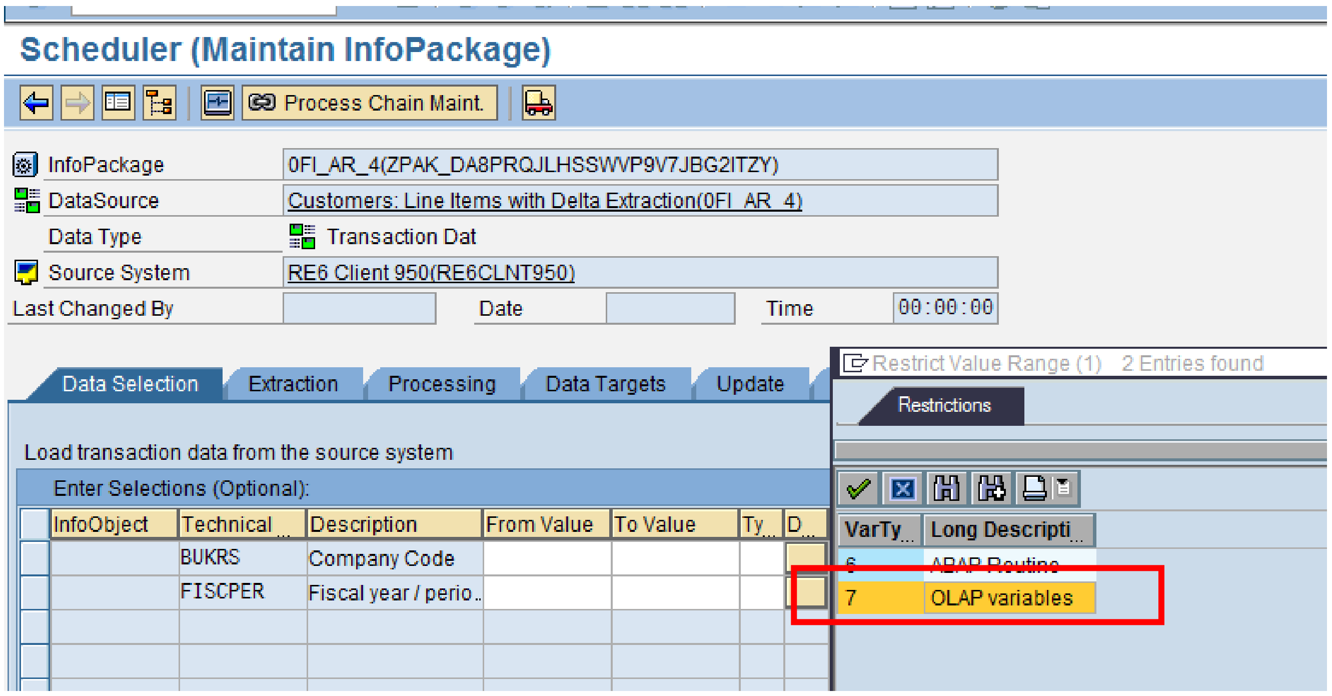
Scheduler (Maintain InfoPackage)

InfoPackage	0FI_AR_4(ZPAK_DA8PRQJLHSSWVP9V7JBG2ITZY)
DataSource	Customers: Line Items with Delta Extraction(0FI_AR_4)
Data Type	Transaction Dat
Source System	RE6 Client 950(RE6CLNT950)
Last Changed By	Date Time 00:00:00

Load transaction data from the source system

Enter Selections (Optional):

InfoObject	Technical ...	Description	From Value	To Value	Ty...	D...	Type (Varia...	R	Data ...	Field ...	Conve...
	BUKRS	Company Code							CHAR	4	
	FISCPER	Fiscal year / perio...							NUMC	9	PERI7


Scheduler (Maintain InfoPackage)

InfoPackage	0FI_AR_4(ZPAK_DA8PRQJLHSSWVP9V7JBG2ITZY)
DataSource	Customers: Line Items with Delta Extraction(0FI_AR_4)
Data Type	Transaction Dat
Source System	RE6 Client 950(RE6CLNT950)
Last Changed By	
Date	
Time	00:00:00

Load transaction data from the source system

Enter Selections (Optional):

InfoObject	Technical ...	Description	From Value	To Value	Ty...	D...
	BUKRS	Company Code				
	FISCPER	Fiscal year / perio..				

Restrict Value Range (1) 2 Entries found

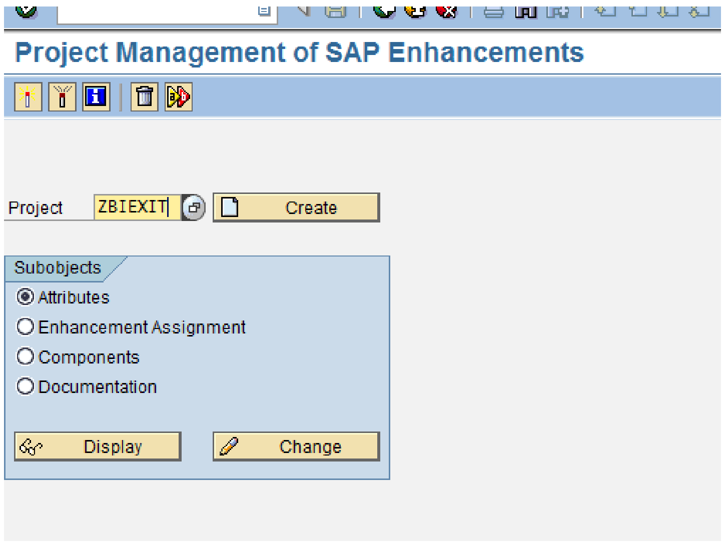
VarTy...	Long Descripti...
6	ABAP Routine
7	OLAP variables

A Pop up appears where we can select the name of the OLAP Variable. Here the name of the variable should be mentioned.

Steps for Creating the OLAP Variable

1) Create a new project using CMOD Transaction code.
2) Select SAP enhancement RSR00001 and assign it to the project.
3) Code the logic.
4) Creation of Variable in BEX Designer.

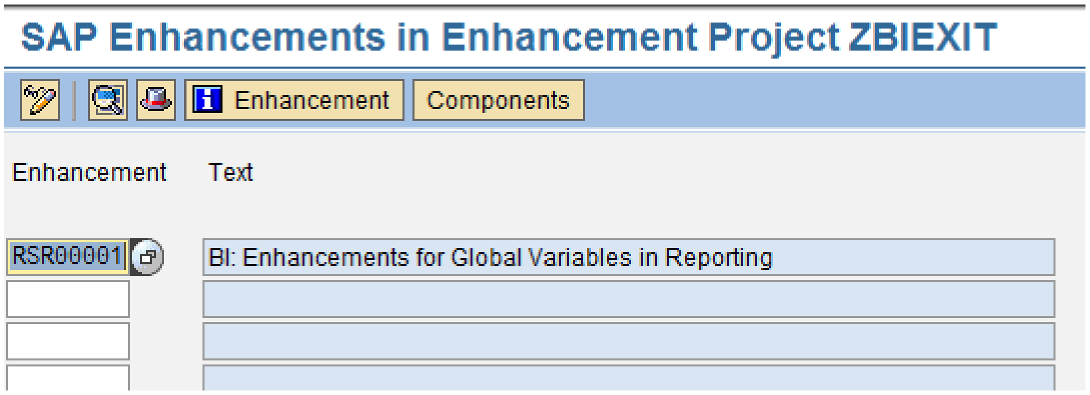
Step 1: Create a new project using CMOD Transaction code.


Project Management of SAP Enhancements

Project ZBIEXIT

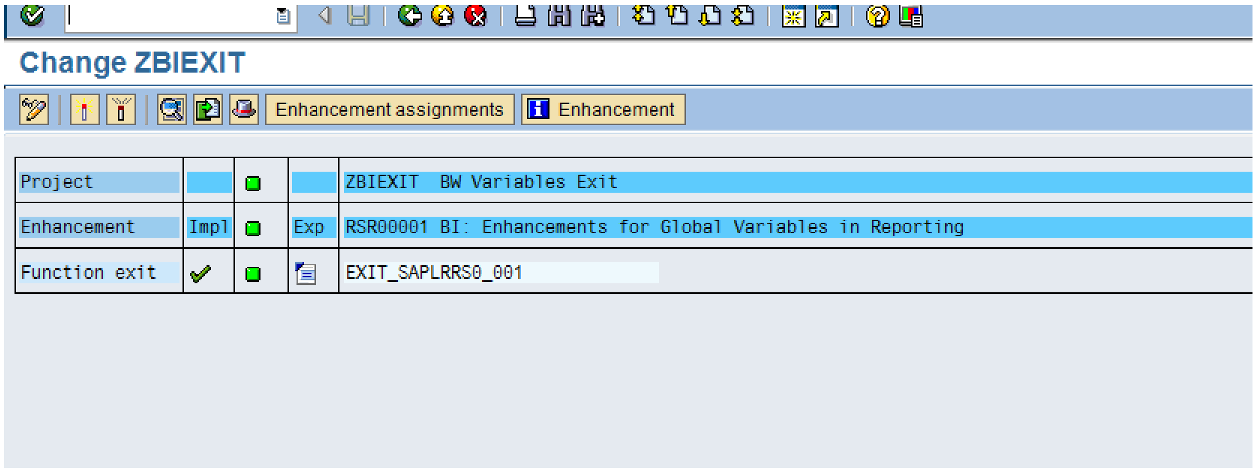
Subobjects
[✓ Attributes]
Enhancement Assignment
Components
Documentation

Step 2: Select SAP enhancement **RSR00001** and assign it to the project.


SAP Enhancements in Enhancement Project ZBIEXIT

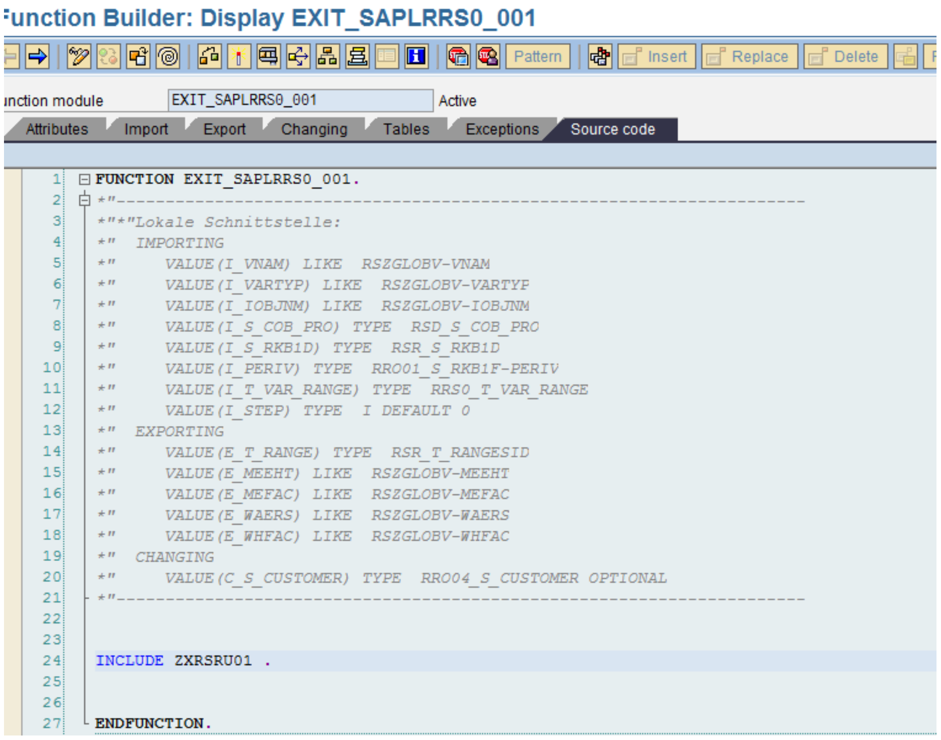
Enhancement	Text
RSR00001	BI: Enhancements for Global Variables in Reporting

In the Components Screen, there is an Exit available and the name of the Exit is '**EXIT_SAPLRRS0_001**', and we can write the code as per requirement.


Change ZBIEXIT

Project		🟩		ZBIEXIT BW Variables Exit
Enhancement	Impl	🟩	Exp	RSR00001 BI: Enhancements for Global Variables in Reporting
Function exit	✓	🟩		EXIT_SAPLRRS0_001

Step 3: Once we click on the exit, we can see the below screen.


Function Builder: Display EXIT_SAPLRRS0_001

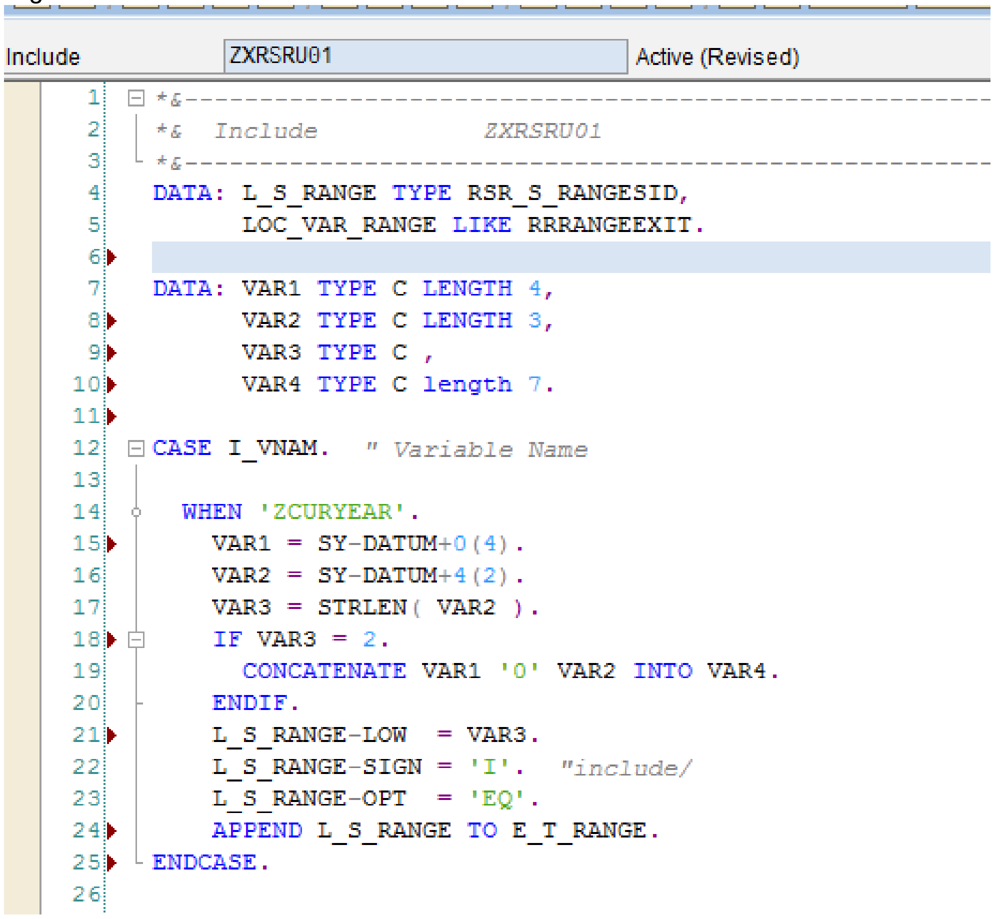
Function module EXIT_SAPLRRS0_001 Active

```
FUNCTION EXIT_SAPLRRS0_001.
*"----------------------------------------------------------------------
*"*"Lokale Schnittstelle:
*"  IMPORTING
*"     VALUE(I_VNAM) LIKE  RSZGLOBV-VNAM
*"     VALUE(I_VARTYP) LIKE  RSZGLOBV-VARTYP
*"     VALUE(I_IOBJNM) LIKE  RSZGLOBV-IOBJNM
*"     VALUE(I_S_COB_PRO) TYPE  RSD_S_COB_PRO
*"     VALUE(I_S_RKB1D) TYPE  RSR_S_RKB1D
*"     VALUE(I_PERIV) TYPE  RRO01_S_RKB1F-PERIV
*"     VALUE(I_T_VAR_RANGE) TYPE  RRS0_T_VAR_RANGE
*"     VALUE(I_STEP) TYPE  I DEFAULT 0
*"  EXPORTING
*"     VALUE(E_T_RANGE) TYPE  RSR_T_RANGESID
*"     VALUE(E_MEEHT) LIKE  RSZGLOBV-MEEHT
*"     VALUE(E_MEFAC) LIKE  RSZGLOBV-MEFAC
*"     VALUE(E_WAERS) LIKE  RSZGLOBV-WAERS
*"     VALUE(E_WHFAC) LIKE  RSZGLOBV-WHFAC
*"  CHANGING
*"     VALUE(C_S_CUSTOMER) TYPE  RRO04_S_CUSTOMER OPTIONAL
*"----------------------------------------------------------------------


INCLUDE ZXRSRU01 .


ENDFUNCTION.
```

Double click on the Include, ABAP Editor appears where the logic can written.

In this example, the value for the field Fiscal Year (FISCPER) should be current fiscal year which needs to be populated from Current System Date (SY-DATUM).

Logic for the Code



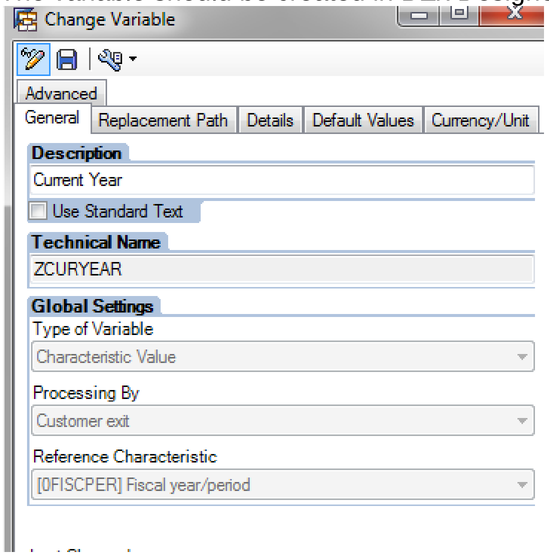
Include ZXRSRU01 Active (Revised)

```
*&---------------------------------------------------------------------
*&  Include           ZXRSRU01
*&---------------------------------------------------------------------
DATA: L_S_RANGE TYPE RSR_S_RANGESID,
      LOC_VAR_RANGE LIKE RRRANGEEXIT.

DATA: VAR1 TYPE C LENGTH 4,
      VAR2 TYPE C LENGTH 3,
      VAR3 TYPE C ,
      VAR4 TYPE C length 7.

CASE I_VNAM.   " Variable Name

  WHEN 'ZCURYEAR'.
    VAR1 = SY-DATUM+0(4).
    VAR2 = SY-DATUM+4(2).
    VAR3 = STRLEN( VAR2 ).
    IF VAR3 = 2.
      CONCATENATE VAR1 '0' VAR2 INTO VAR4.
    ENDIF.
    L_S_RANGE-LOW  = VAR3.
    L_S_RANGE-SIGN = 'I'.  "include/
    L_S_RANGE-OPT  = 'EQ'.
    APPEND L_S_RANGE TO E_T_RANGE.
ENDCASE.
```

Once the logic is written, the code needs to saved and Activate.

Step 4: Creation of Variable in BEX Designer

The variable should be created in BEx Designer.



Description: Current Year

Use Standard Text: [ ]

Technical Name: ZCURYEAR

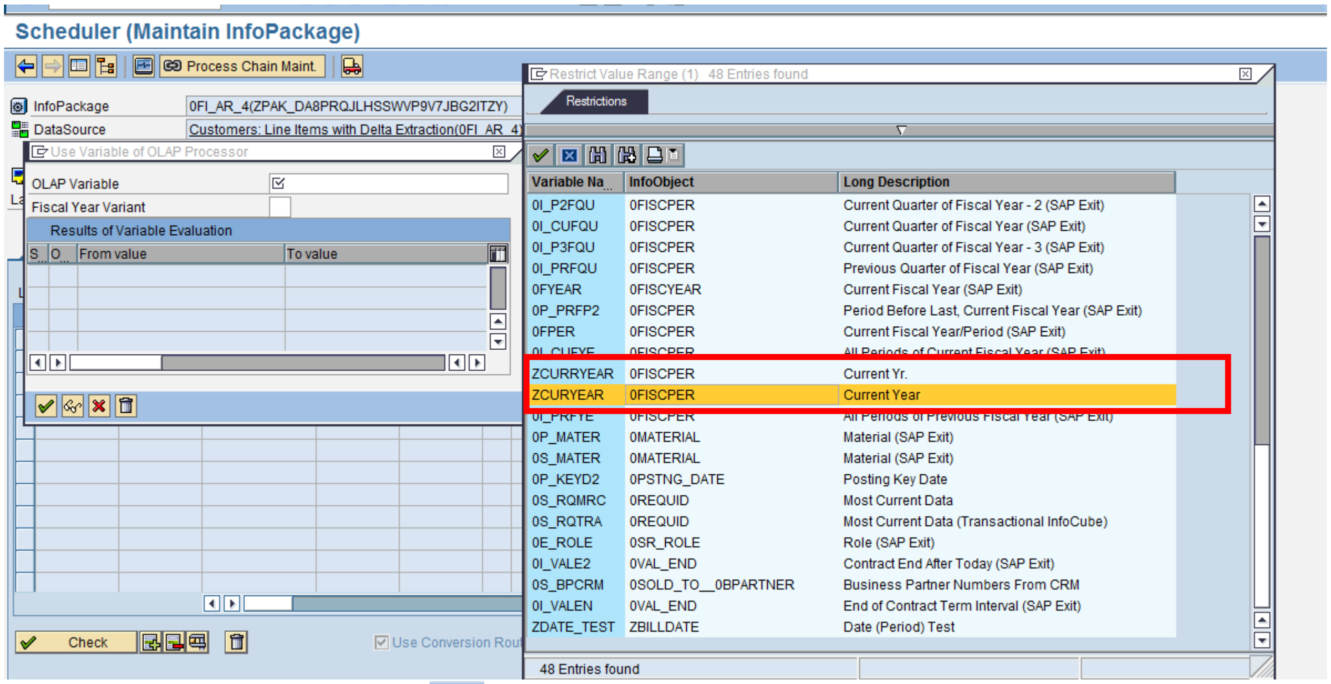
Global Settings

Type of Variable: Characteristic Value

Processing By: Customer exit

Reference Characteristic: [0FISCPER] Fiscal year/period

Once Variable creation is done, the variable will appear the f4 help in the InfoPackage OLAP Variable screen.



Scheduler (Maintain InfoPackage)

Process Chain Maint.

InfoPackage 0FI_AR_4(ZPAK_DA8PRQJLHSSWVP9V7JBG2ITZY)
DataSource Customers: Line Items with Delta Extraction(0FI_AR_4)

Use Variable of OLAP Processor

OLAP Variable
Fiscal Year Variant

Results of Variable Evaluation

S..	O...	From value	To value

Check

Use Conversion Rout

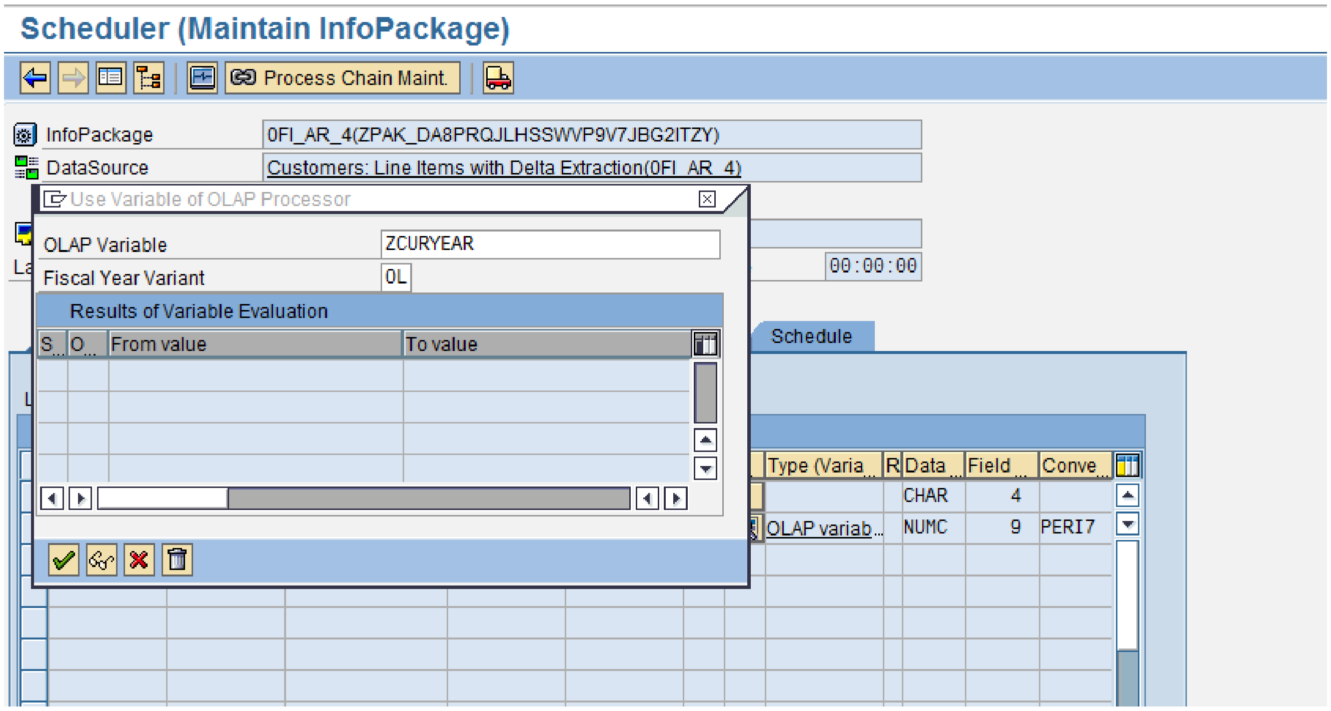
Restrict Value Range (1) 48 Entries found

Restrictions

Variable Na...	InfoObject	Long Description
0I_P2FQU	0FISCPER	Current Quarter of Fiscal Year - 2 (SAP Exit)
0I_CUFQU	0FISCPER	Current Quarter of Fiscal Year (SAP Exit)
0I_P3FQU	0FISCPER	Current Quarter of Fiscal Year - 3 (SAP Exit)
0I_PRFQU	0FISCPER	Previous Quarter of Fiscal Year (SAP Exit)
0FYEAR	0FISCYEAR	Current Fiscal Year (SAP Exit)
0P_PRFP2	0FISCPER	Period Before Last, Current Fiscal Year (SAP Exit)
0FPER	0FISCPER	Current Fiscal Year/Period (SAP Exit)
0I_CUFYE	0FISCPER	All Periods of Current Fiscal Year (SAP Exit)
ZCURRYEAR	0FISCPER	Current Yr.
ZCURYEAR	0FISCPER	Current Year
0I_PRFYE	0FISCPER	All Periods of Previous Fiscal Year (SAP Exit)
0P_MATER	0MATERIAL	Material (SAP Exit)
0S_MATER	0MATERIAL	Material (SAP Exit)
0P_KEYD2	0PSTNG_DATE	Posting Key Date
0S_RQMRC	0REQUID	Most Current Data
0S_RQTRA	0REQUID	Most Current Data (Transactional InfoCube)
0E_ROLE	0SR_ROLE	Role (SAP Exit)
0I_VALE2	0VAL_END	Contract End After Today (SAP Exit)
0S_BPCRM	0SOLD_TO__0BPARTNER	Business Partner Numbers From CRM
0I_VALEN	0VAL_END	End of Contract Term Interval (SAP Exit)
ZDATE_TEST	ZBILLDATE	Date (Period) Test

48 Entries found

Select the variable and select O.K. ( )



Scheduler (Maintain InfoPackage)

Process Chain Maint.

InfoPackage 0FI_AR_4(ZPAK_DA8PRQJLHSSWVP9V7JBG2ITZY)
DataSource Customers: Line Items with Delta Extraction(0FI_AR_4)

Use Variable of OLAP Processor

OLAP Variable ZCURYEAR
Fiscal Year Variant 0L

Results of Variable Evaluation

S..	O...	From value	To value

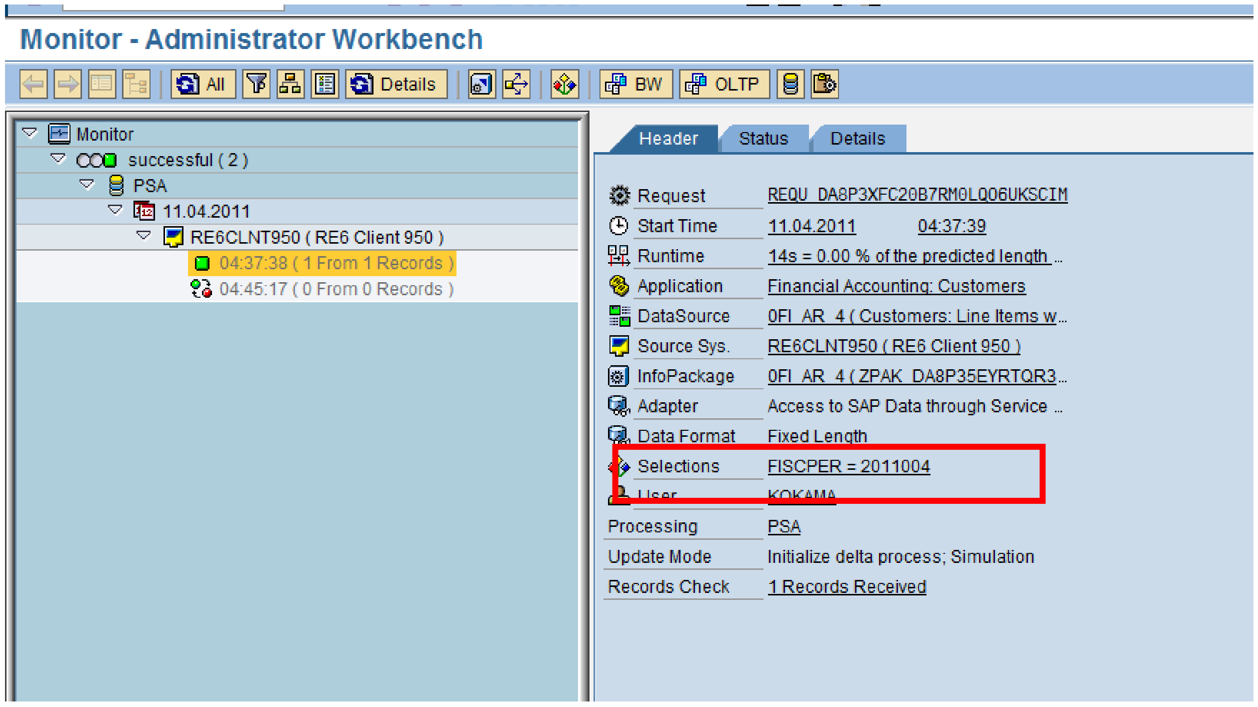
00:00:00

Schedule

Type (Varia...	R	Data ...	Field ...	Conve...
		CHAR	4	
OLAP variab...		NUMC	9	PERI7

Save the InfoPackage and schedule it.

In the Monitor-Header screen the selections can be seen.



Monitor - Administrator Workbench

All
Details
BW
OLTP
Monitor
successful (2)
PSA
11.04.2011
RE6CLNT950 (RE6 Client 950)
04:37:38 (1 From 1 Records)
04:45:17 (0 From 0 Records)
Header
Status
Details

Request	REQU_DA8P3XFC20B7RM0LQ06UKSCIM
Start Time	11.04.2011 04:37:39
Runtime	14s = 0.00 % of the predicted length ...
Application	Financial Accounting: Customers
DataSource	0FI_AR_4 (Customers: Line Items w...
Source Sys.	RE6CLNT950 (RE6 Client 950)
InfoPackage	0FI_AR_4 (ZPAK_DA8P35EYRTQR3...
Adapter	Access to SAP Data through Service ...
Data Format	Fixed Length
Selections	FISCPER = 2011004
User	KOKAMA
Processing	PSA
Update Mode	Initialize delta process; Simulation
Records Check	1 Records Received

DTP Filter:

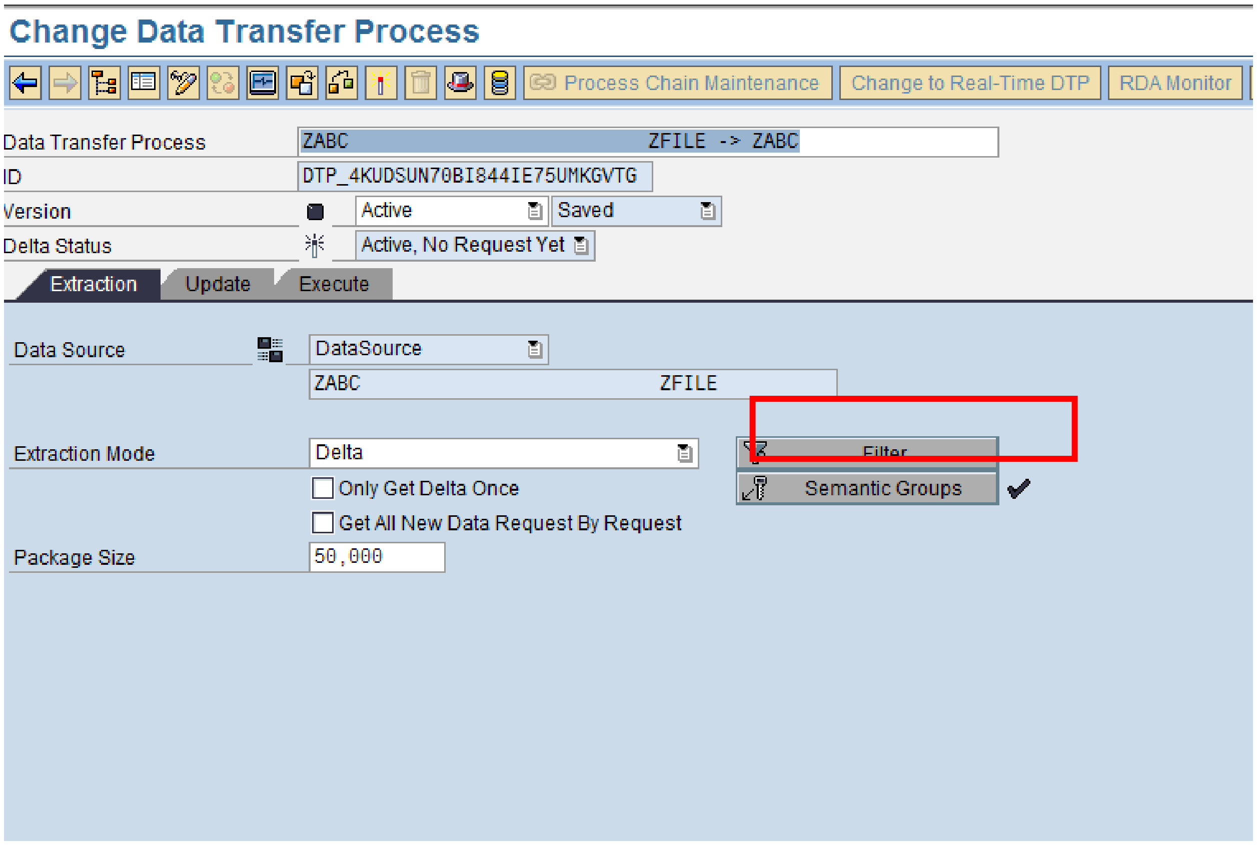
The data transfer process (DTP) is used to transfer data from source objects to target objects in BI. It can also be used to access InfoProvider data directly.

In the extraction tab of the DSO, the Filter Tab can be seen. The filter criteria can be determined for the data using the Filter function.

Multiple data transfer processes can be used 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. The filter thus restricts the amount of data to be copied and works like the selections in the InfoPackage. Single values, multiple selections, intervals, selections based on variables, or routines can also be specified.

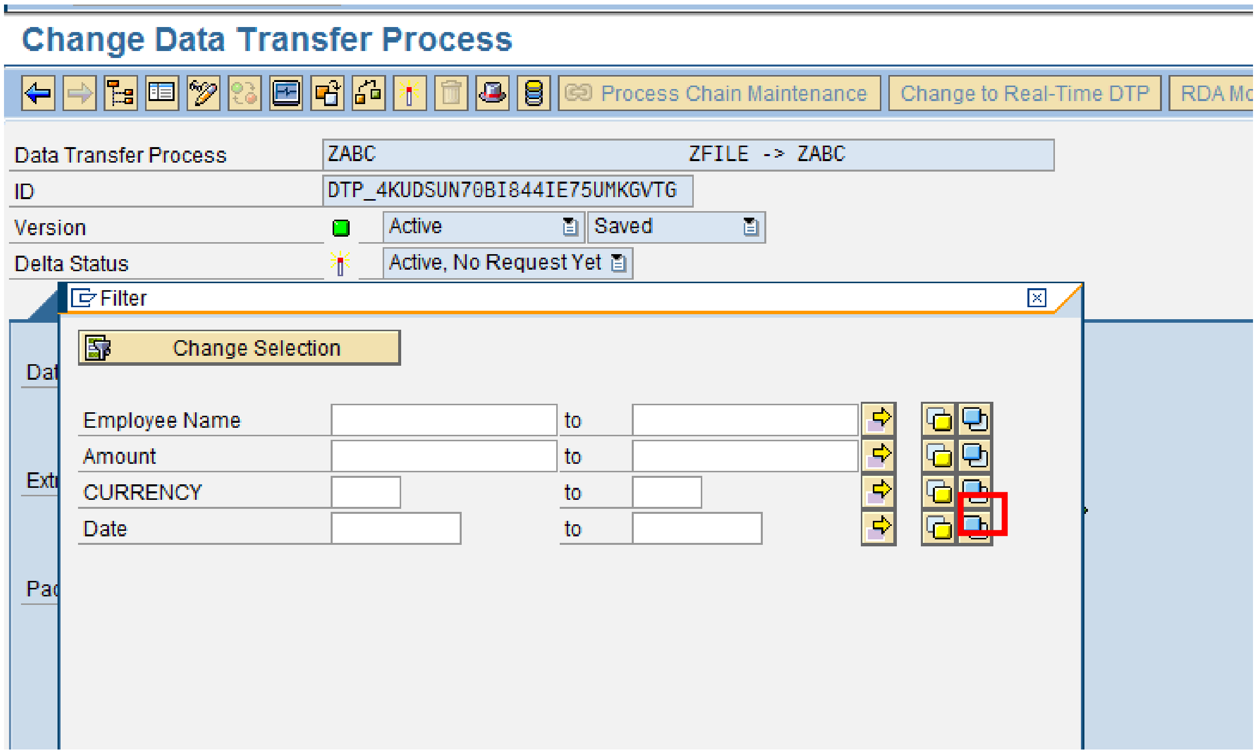
Example:

The below example is demonstrated by restricting the data for the filed ‘0DATE’.



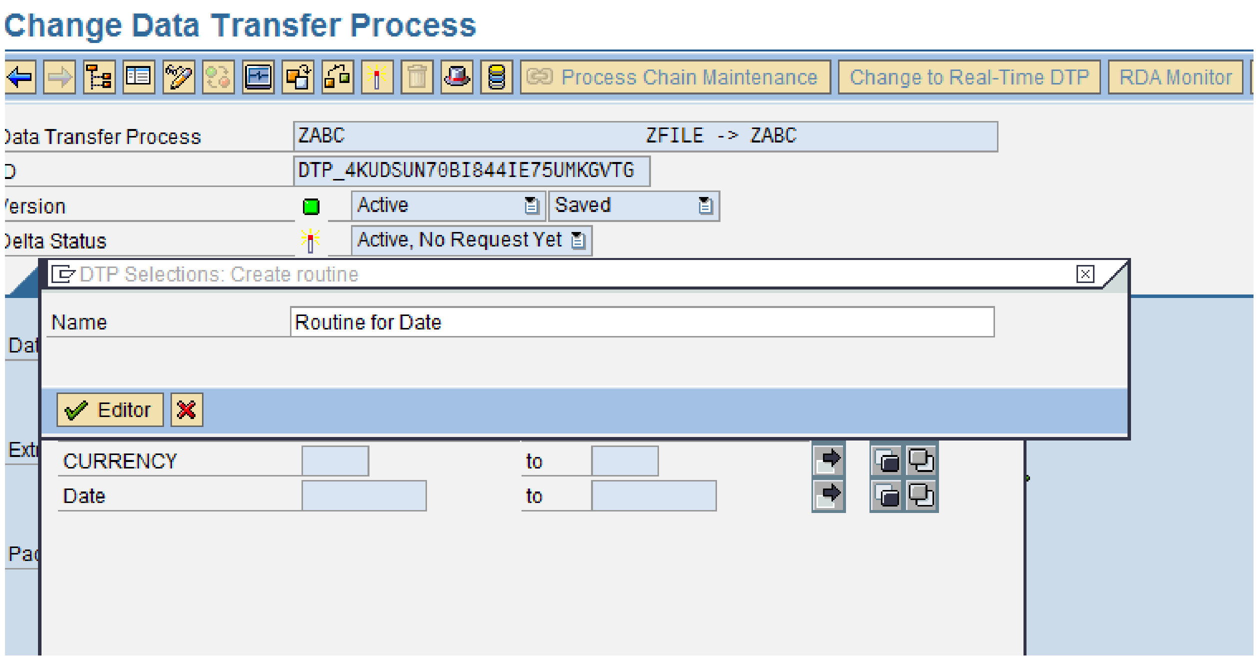
Data Transfer Process	ZABC	ZFILE -> ZABC
ID	DTP_4KUDSUN70BI844IE75UMKGVTG	
Version	Active	Saved
Delta Status	Active, No Request Yet	
Data Source	DataSource	
	ZABC	ZFILE
Extraction Mode	Delta	
	☐ Only Get Delta Once	
	☐ Get All New Data Request By Request	
Package Size	50,000	

Click on the Blue Icon (ABAP Routine) next as shown



Data Transfer Process	ZABC	ZFILE -> ZABC
ID	DTP_4KUDSUN70BI844IE75UMKGVTG	
Version	Active	Saved
Delta Status	Active, No Request Yet	

Employee Name		to	
Amount		to	
CURRENCY		to	
Date		to	


Change Data Transfer Process

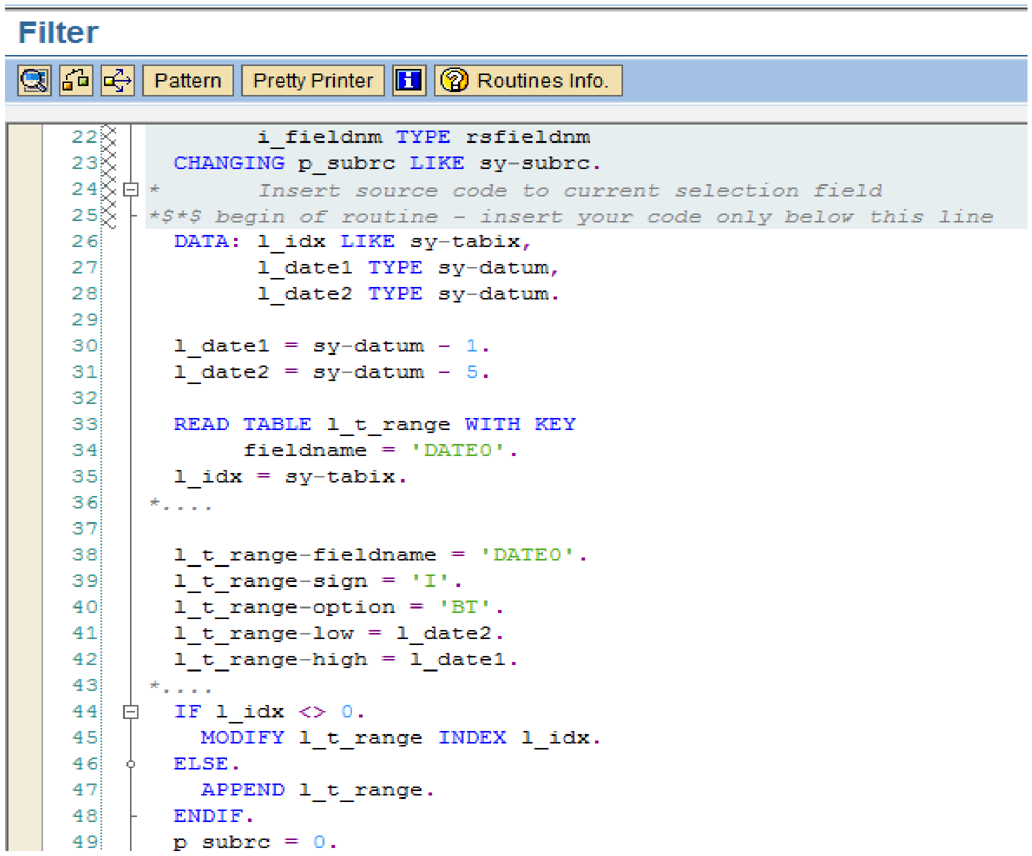
Data Transfer Process: ZABC ZFILE -> ZABC
ID: DTP_4KUDSUN70BI844IE75UMKGVTG
Version: Active Saved
Delta Status: Active, No Request Yet

DTP Selections: Create routine

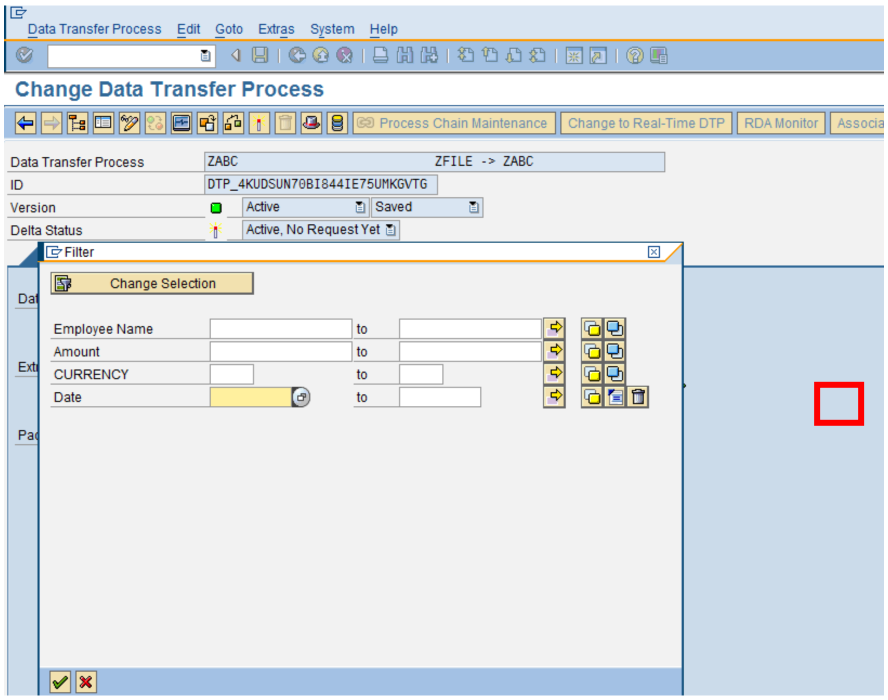
Name: Routine for Date

CURRENCY to
Date to

Enter the source code in the ABAP Editor


Filter

```
          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: l_idx LIKE sy-tabix,
          l_date1 TYPE sy-datum,
          l_date2 TYPE sy-datum.

  l_date1 = sy-datum - 1.
  l_date2 = sy-datum - 5.

  READ TABLE l_t_range WITH KEY
       fieldname = 'DATE0'.
  l_idx = sy-tabix.
*....

  l_t_range-fieldname = 'DATE0'.
  l_t_range-sign = 'I'.
  l_t_range-option = 'BT'.
  l_t_range-low = l_date2.
  l_t_range-high = l_date1.
*....
  IF l_idx <> 0.
    MODIFY l_t_range INDEX l_idx.
  ELSE.
    APPEND l_t_range.
  ENDIF.
  p_subrc = 0.
```

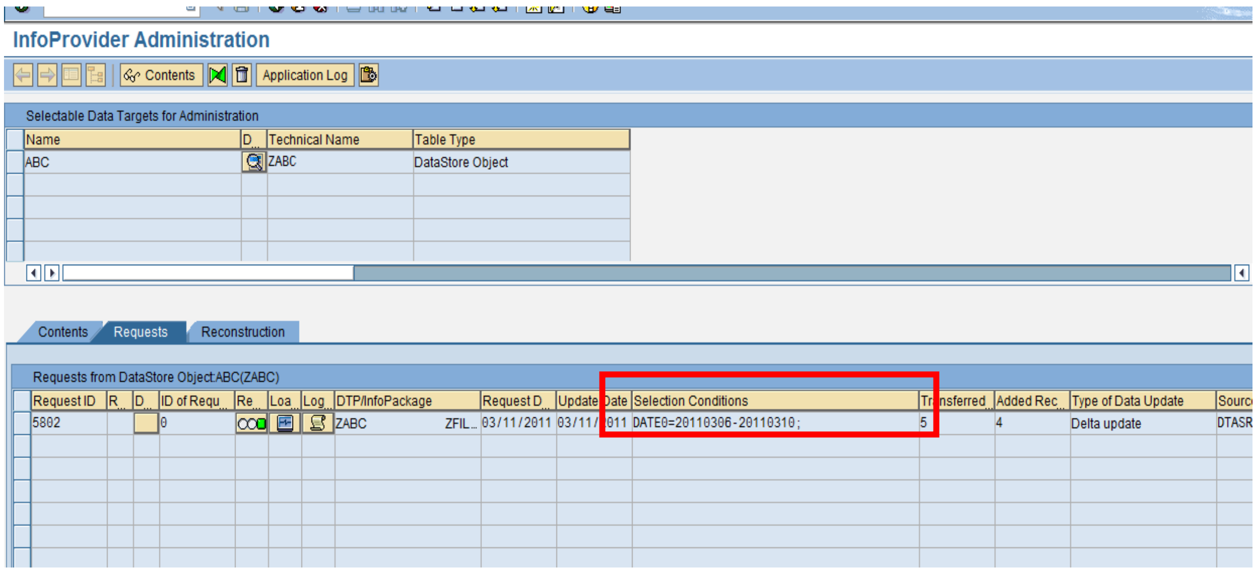


Data Transfer Process	ZABC	ZFILE -> ZABC
ID	DTP_4KUDSUN70BI844IE75UMKGVTG	
Version	Active	Saved
Delta Status	Active, No Request Yet	

		to	
Employee Name		to	
Amount		to	
CURRENCY		to	
Date		to	

Activate the DTP.

After executing the DTP, we can check the selection conditions in the DSO Request Tab.



Selectable Data Targets for Administration

Name	D	Technical Name	Table Type
ABC		ZABC	DataStore Object

Requests from DataStore Object:ABC(ZABC)

Request ID	R...	D...	ID of Requ...	Re...	Loa...	Log...	DTP/InfoPackage	Request D...	Update Date	Selection Conditions	Transferred	Added Rec...	Type of Data Update	Sourc
5802			0				ZABC ZFIL...	03/11/2011	03/11/2011	DATE0=20110306-20110310;	5	4	Delta update	DTASR

Routines in Transformations

The transformation process allows you to consolidate, cleanse, and integrate data.

When the data is loaded from one BI object into a further BI object, the data is passed through a transformation in the form of packets. A transformation converts the fields of the source into the format of the target.

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.

Transformation type
The transformation type determines how data is written into the fields of the target.

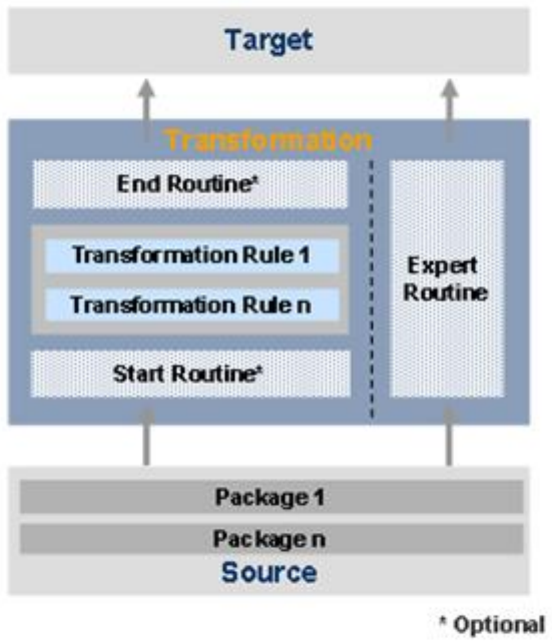
Rule group
A rule group is a group of transformation rules. Rule groups allow you to combine various rules.

Routine

Routines are used to implement complex transformation rules. Routines are available as a rule type. There are also routine types that you can use to implement additional transformations.

For every data packet, the Start transformation is executed first, then the Transformation rules area executed and finally end Routine will be executed.

The following graph explains the same.



<u>Types of Routines</u>

Routines are mainly used in transformation for manipulating and transforming the data according to the user’s requirement.

For example, if we have to add two quantity fields which are in source and populate the result into a single field in the target, then routines can be used to accomplish this scenario.

Three types of Routines which are used in transformations are:

1) Start Routine
2) End Routine
3) Expert Routine
4) Routine for Characteristics or Key Figures.

Features:

The routine has a global part and a local part. In the global part the global data declarations can be defined. 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.

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.

Start Routine Parameters:

From the Source object to the target object, the data is transferred in the form of packets.

Importing

REQUEST: Request ID

DATAPAKID: Number of current data package

The input for start is Request ID and Data Package number of the current packet data.

The Internal table which contains data is 'SOURCE_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.

End Routine

An end routine is a routine with a table in the target structure format. You can use an end routine to postprocess data after transformation on a package-by-package basis. It is triggered after Transformation.

End Routine Parameters:

From the Source object to the target object, the data is transferred in the form of packets.

Importing

REQUEST: Request ID

DATAPAKID: Number of current data package

The Internal table which contains data is 'RESULT_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

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).

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.

Routine for Characteristics or Key Figures

A routine can be created for updating or modifying a single characteristic or key figure.

Routine Parameters:

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: Assign the result of the computed key figure or computed characteristic to the RESULT Variable.
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.

Expert Routine

This type of routine is only intended for use in special cases. The expert routine can be used 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.

This can be used to program the transformation without using available rule types. An expert routine performs all the actions of Start Routine, End routine and Field Routine. All the rules of transformation need to be coded in the Expert Routine.

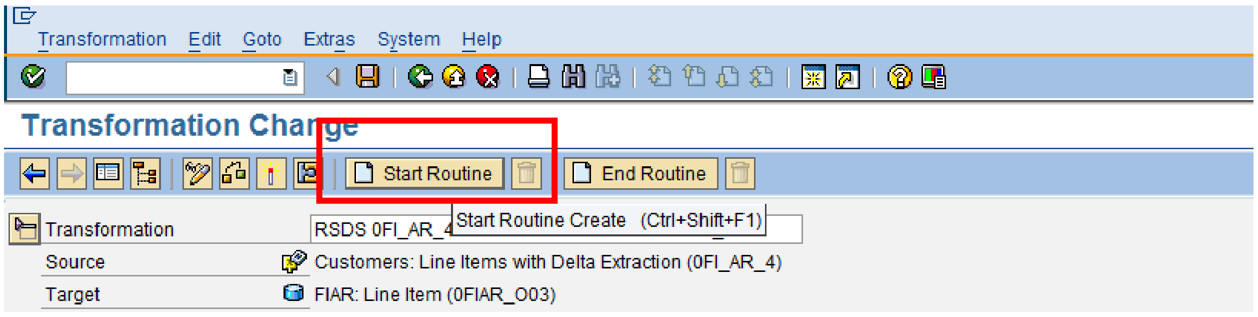
Expert routine has internal table SOURCE_PACKAGE which has all the source data, and need to be manipulated and transferred into Internal Table RESULT_PACKAGE, which is of the target structure.

Note that if you have already created transformation rules, the system deletes them once you have created an expert routine and if the target of the transformation is a DataStore object, key figures are updated by default with the aggregation behavior Overwrite (MOVE).

Example for Start Routine, End Routine Field Routine and Expert Routine

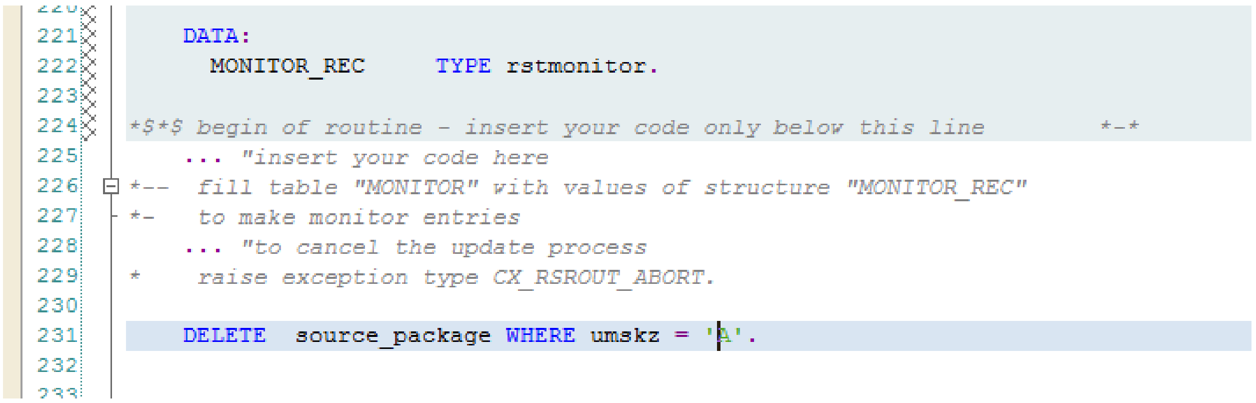
Start Routine:

In the transformation, click on Start Routine as shown in below diagram.


Transformation Edit Goto Extras System Help
Transformation Change
Start Routine
End Routine
Start Routine Create (Ctrl+Shift+F1)

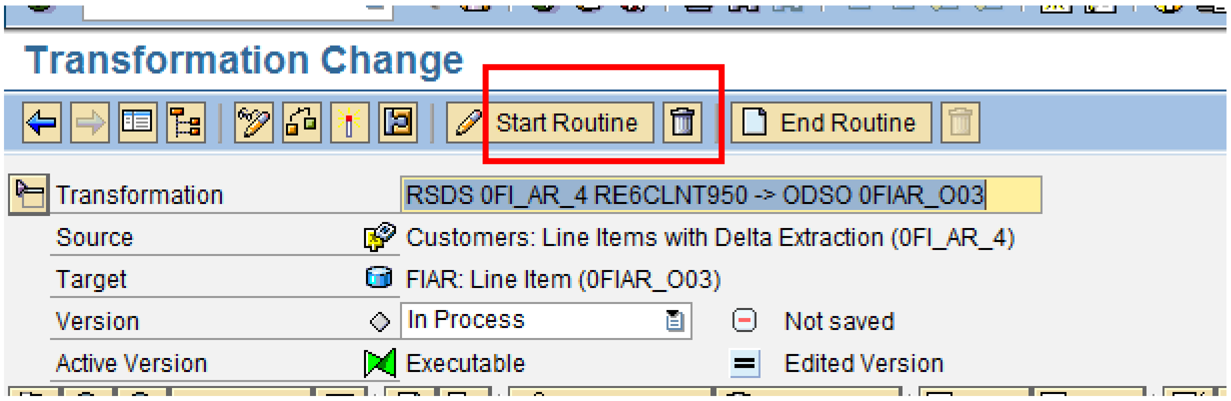
Transformation RSDS 0FI_AR_4
Source Customers: Line Items with Delta Extraction (0FI_AR_4)
Target FIAR: Line Item (0FIAR_O03)

In the Source code section, the code can be written as per the requirement. In this case there is a requirement to delete certain records from the Source data.



```
    DATA:
      MONITOR_REC     TYPE rstmonitor.

*$*$ begin of routine - insert your code only below this line        *-*
    ... "insert your code here
*--  fill table "MONITOR" with values of structure "MONITOR_REC"
*-   to make monitor entries
    ... "to cancel the update process
*    raise exception type CX_RSROUT_ABORT.

    DELETE  source_package WHERE umskz = 'A'.
```

In this case, the data from source data will be deleted where the UMSKZ (Special G/L ind) = ‘A’. Save the routine, once the coding is done.

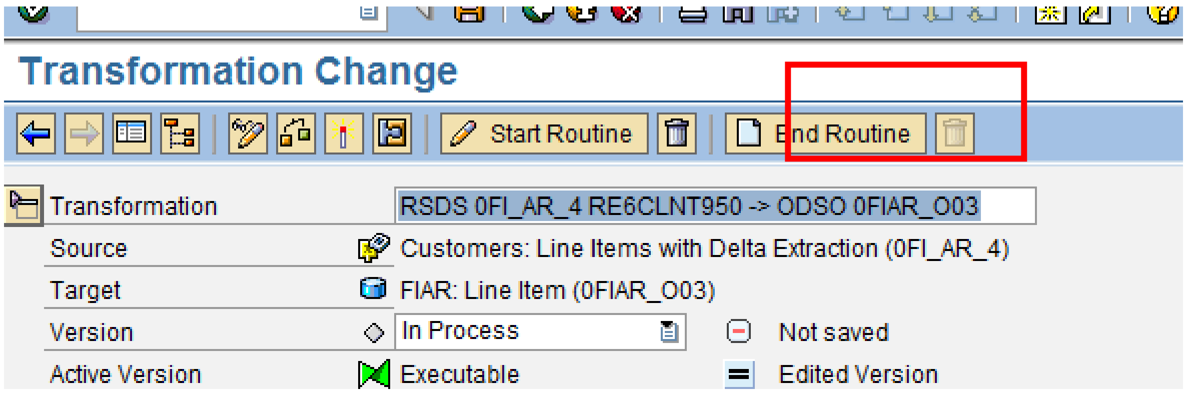


Transformation RSDS 0FI_AR_4 RE6CLNT950 -> ODSO 0FIAR_O03
Source Customers: Line Items with Delta Extraction (0FI_AR_4)
Target FIAR: Line Item (0FIAR_O03)
Version In Process Not saved
Active Version Executable Edited Version

As shown above, the start routine is created.

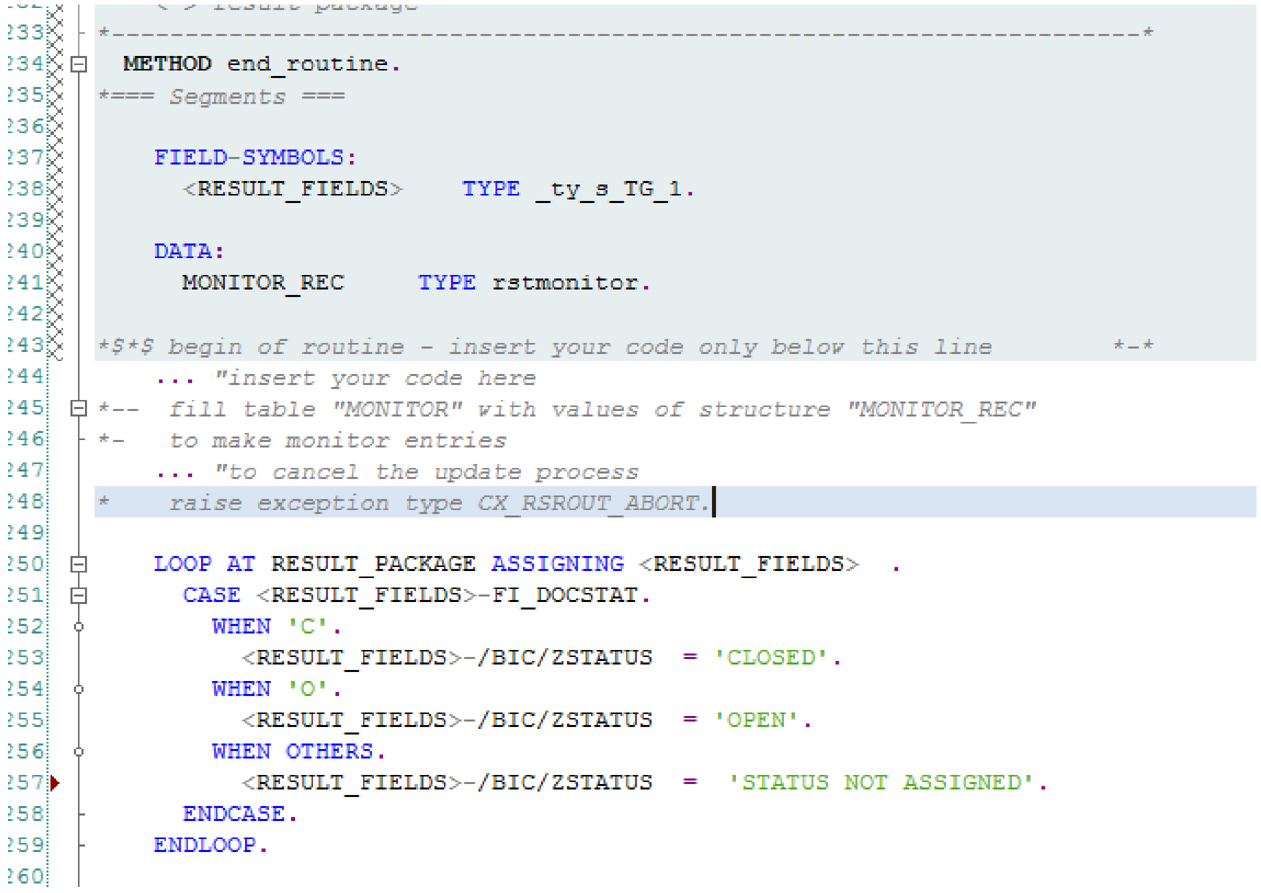
End Routine

Click on end routine, to create the end routine.



Transformation	RSDS 0FI_AR_4 RE6CLNT950 -> ODSO 0FIAR_O03		
Source	Customers: Line Items with Delta Extraction (0FI_AR_4)		
Target	FIAR: Line Item (0FIAR_O03)		
Version	In Process		Not saved
Active Version	Executable		Edited Version

In the Source code section, the code can be written as per the requirement. In this case, there is requirement to populate a new filed with text, based on the data from source.

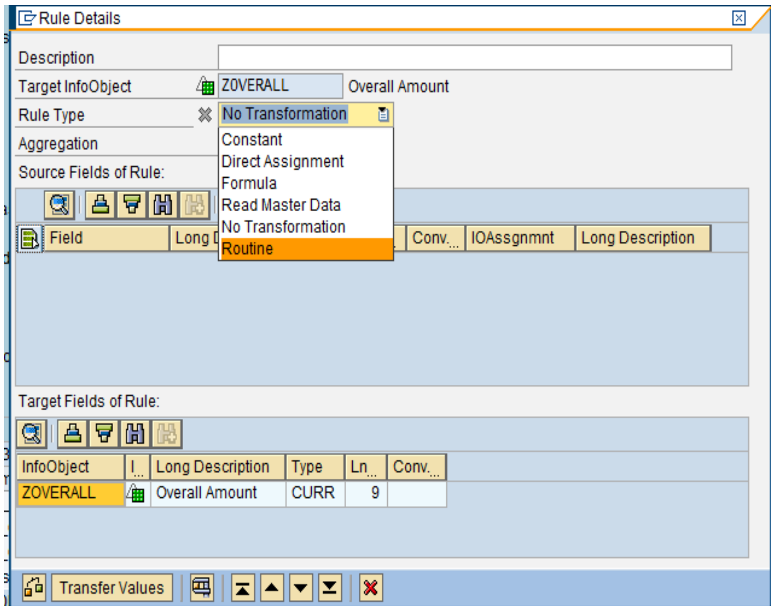


```
*----------------------------------------------------------------------*
  METHOD end_routine.
*=== Segments ===

    FIELD-SYMBOLS:
      <RESULT_FIELDS>    TYPE _ty_s_TG_1.

    DATA:
      MONITOR_REC     TYPE rstmonitor.

*$*$ begin of routine - insert your code only below this line        *-*
    ... "insert your code here
*--  fill table "MONITOR" with values of structure "MONITOR_REC"
*-   to make monitor entries
    ... "to cancel the update process
*    raise exception type CX_RSROUT_ABORT.

    LOOP AT RESULT_PACKAGE ASSIGNING <RESULT_FIELDS>   .
      CASE <RESULT_FIELDS>-FI_DOCSTAT.
        WHEN 'C'.
          <RESULT_FIELDS>-/BIC/ZSTATUS   = 'CLOSED'.
        WHEN 'O'.
          <RESULT_FIELDS>-/BIC/ZSTATUS   = 'OPEN'.
        WHEN OTHERS.
          <RESULT_FIELDS>-/BIC/ZSTATUS   =  'STATUS NOT ASSIGNED'.
      ENDCASE.
    ENDLOOP.
```

In this case we are checking the field FI_DOCSTAT. If FI_DOCSTAT = C then the field ZSTATUS will be populated as ‘CLOSED’. If FI_DOCSTAT = O then the field ZSTATUS will be populated as ‘OPEN’ and FI_DOCSTAT has value anything other than ‘C’ & ‘O’ then the field ZSTATUS will be populated as ‘STATUS NOT ASSIGNED’.

In the code, the RESULT_PACKAGE (This Internal table contains the data that needs to be populated finally into the target) is looped into <RESULT_FIELDS> field symbol and each record is changed as required.

Field Routine or Routine for Characteristics or Key Figures



Rule Details

Description	
Target InfoObject	ZOVERALL Overall Amount
Rule Type	No Transformation
Aggregation	

Source Fields of Rule:

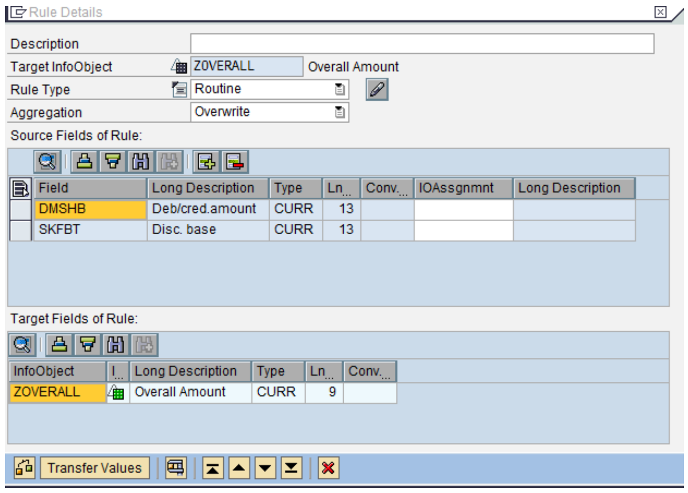
Field	Long D	Conv.	IOAssgnmnt	Long Description

Target Fields of Rule:

InfoObject	I...	Long Description	Type	Ln...	Conv.
ZOVERALL		Overall Amount	CURR	9	

Click on the InfoObject, where the field routine is to be created, and define the Rule Type as ‘Routine’.

Add the InfoObjects that are required for the calculation in the routine. In this case we are doing calculations on two Key Figures and populating the result to another Key Figure. Once we click the rule type as ‘Routine’ ABAP editor will be displayed, where the coding can be done.



Rule Details

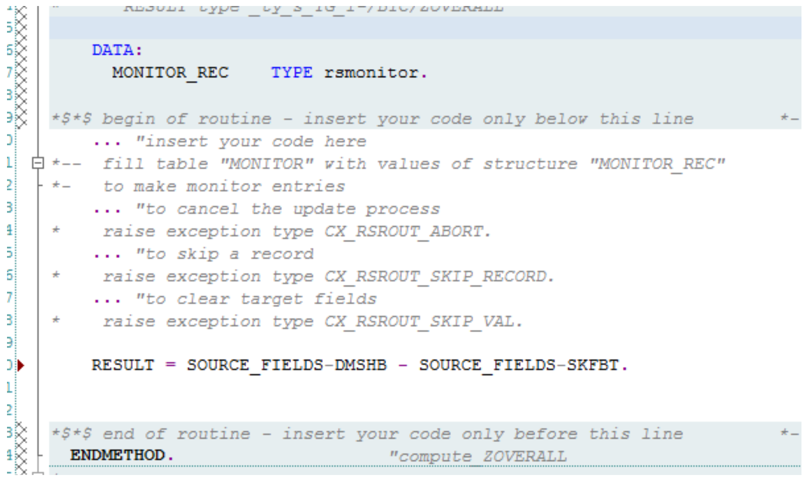
Description	
Target InfoObject	ZOVERALL Overall Amount
Rule Type	Routine
Aggregation	Overwrite

Source Fields of Rule:

Field	Long Description	Type	Ln...	Conv...	IOAssgnmnt	Long Description
DMSHB	Deb/cred.amount	CURR	13			
SKFBT	Disc. base	CURR	13			

Target Fields of Rule:

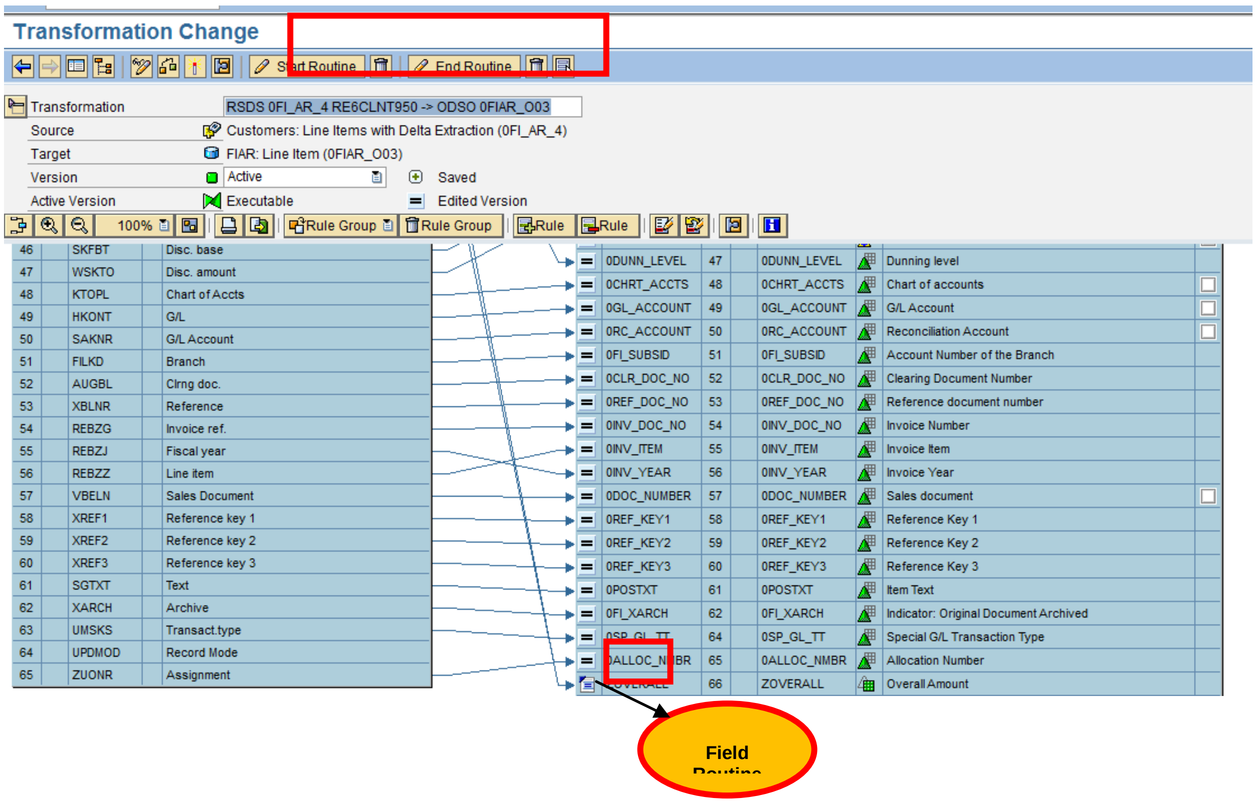
InfoObject	I...	Long Description	Type	Ln...	Conv...
ZOVERALL		Overall Amount	CURR	9	



```
*     RESULT type _ty_s_TG_1-/BIC/ZOVERALL

    DATA:
      MONITOR_REC     TYPE rsmonitor.

*$*$ begin of routine - insert your code only below this line        *-
    ... "insert your code here
*--   fill table "MONITOR" with values of structure "MONITOR_REC"
*-    to make monitor entries
    ... "to cancel the update process
*     raise exception type CX_RSROUT_ABORT.
    ... "to skip a record
*     raise exception type CX_RSROUT_SKIP_RECORD.
    ... "to clear target fields
*     raise exception type CX_RSROUT_SKIP_VAL.

    RESULT = SOURCE_FIELDS-DMSHB - SOURCE_FIELDS-SKFBT.


*$*$ end of routine - insert your code only before this line         *-
  ENDMETHOD.                    "compute_ZOVERALL
```

In this case, we are subtracting the SKNTO (Discount amount) from the DMSHB (Total amount).

Save the routine.



Transformation Change

Transformation	RSDS 0FI_AR_4 RE6CLNT950 -> ODSO 0FIAR_O03
Source	Customers: Line Items with Delta Extraction (0FI_AR_4)
Target	FIAR: Line Item (0FIAR_O03)
Version	Active — Saved
Active Version	Executable — Edited Version

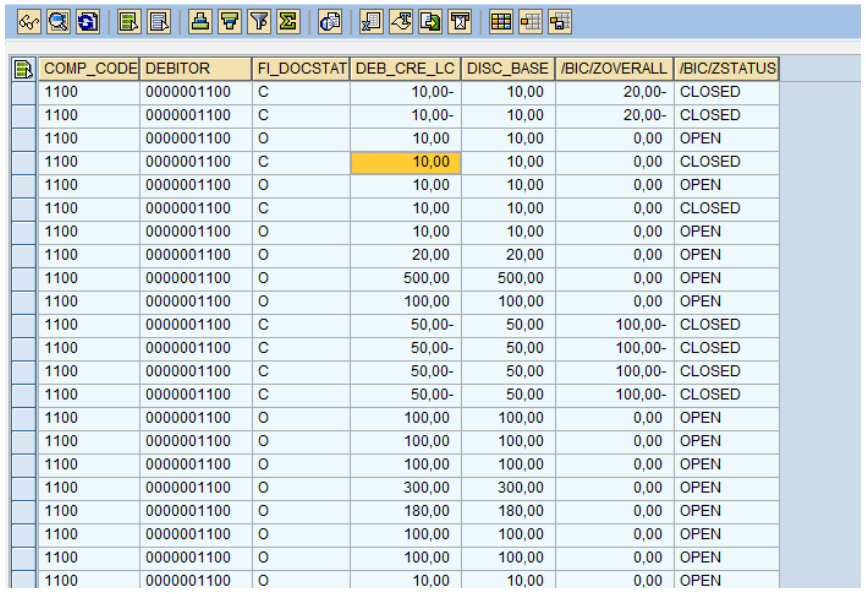
46	SKFBT	Disc. base
47	WSKTO	Disc. amount
48	KTOPL	Chart of Accts
49	HKONT	G/L
50	SAKNR	G/L Account
51	FILKD	Branch
52	AUGBL	Clrng doc.
53	XBLNR	Reference
54	REBZG	Invoice ref.
55	REBZJ	Fiscal year
56	REBZZ	Line item
57	VBELN	Sales Document
58	XREF1	Reference key 1
59	XREF2	Reference key 2
60	XREF3	Reference key 3
61	SGTXT	Text
62	XARCH	Archive
63	UMSKS	Transact.type
64	UPDMOD	Record Mode
65	ZUONR	Assignment

0DUNN_LEVEL	47	0DUNN_LEVEL	Dunning level
0CHRT_ACCTS	48	0CHRT_ACCTS	Chart of accounts
0GL_ACCOUNT	49	0GL_ACCOUNT	G/L Account
0RC_ACCOUNT	50	0RC_ACCOUNT	Reconciliation Account
0FI_SUBSID	51	0FI_SUBSID	Account Number of the Branch
0CLR_DOC_NO	52	0CLR_DOC_NO	Clearing Document Number
0REF_DOC_NO	53	0REF_DOC_NO	Reference document number
0INV_DOC_NO	54	0INV_DOC_NO	Invoice Number
0INV_ITEM	55	0INV_ITEM	Invoice Item
0INV_YEAR	56	0INV_YEAR	Invoice Year
0DOC_NUMBER	57	0DOC_NUMBER	Sales document
0REF_KEY1	58	0REF_KEY1	Reference Key 1
0REF_KEY2	59	0REF_KEY2	Reference Key 2
0REF_KEY3	60	0REF_KEY3	Reference Key 3
0POSTXT	61	0POSTXT	Item Text
0FI_XARCH	62	0FI_XARCH	Indicator: Original Document Archived
0SP_GL_TT	64	0SP_GL_TT	Special G/L Transaction Type
0ALLOC_NMBR	65	0ALLOC_NMBR	Allocation Number
ZOVERALL	66	ZOVERALL	Overall Amount

After executing the DTP,

1. The data doesn't contain records where UMSKZ (Special G/L ind) = 'A'.
2. The calculated data will be populated into the field 'ZOVERALL'.
3. As per document status, new InfoObject 'ZTATUS' will be populated with text values.

DSO Output

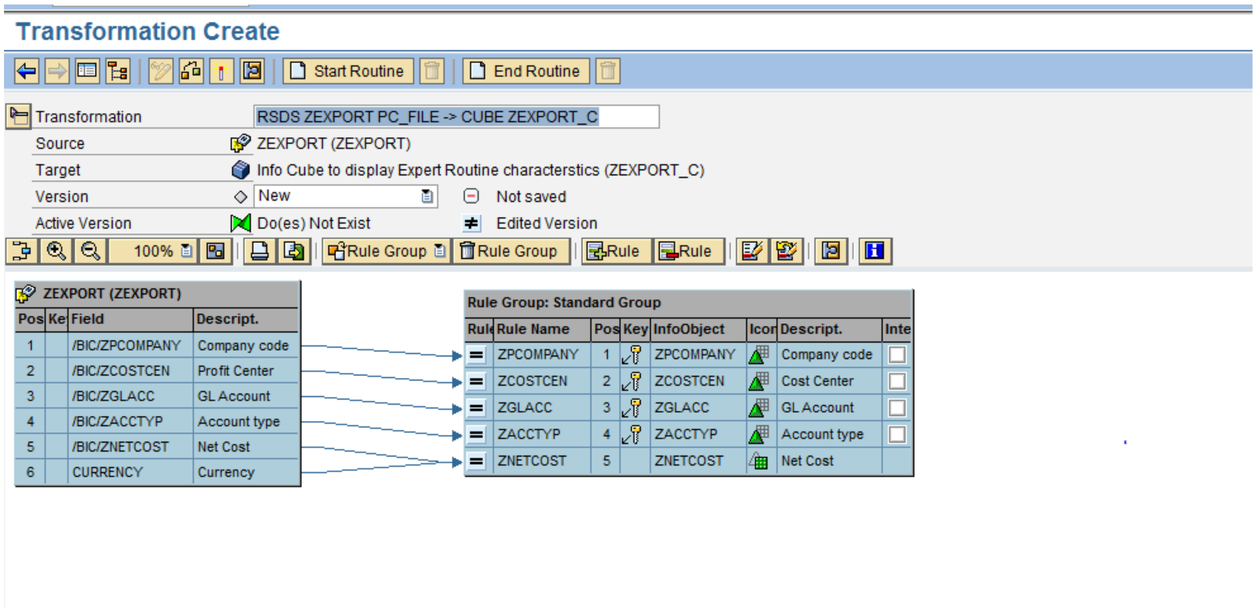


COMP_CODE	DEBITOR	FI_DOCSTAT	DEB_CRE_LC	DISC_BASE	/BIC/ZOVERALL	/BIC/ZSTATUS
1100	0000001100	C	10,00-	10,00	20,00-	CLOSED
1100	0000001100	C	10,00-	10,00	20,00-	CLOSED
1100	0000001100	O	10,00	10,00	0,00	OPEN
1100	0000001100	C	10,00	10,00	0,00	CLOSED
1100	0000001100	O	10,00	10,00	0,00	OPEN
1100	0000001100	C	10,00	10,00	0,00	CLOSED
1100	0000001100	O	10,00	10,00	0,00	OPEN
1100	0000001100	O	20,00	20,00	0,00	OPEN
1100	0000001100	O	500,00	500,00	0,00	OPEN
1100	0000001100	O	100,00	100,00	0,00	OPEN
1100	0000001100	C	50,00-	50,00	100,00-	CLOSED
1100	0000001100	C	50,00-	50,00	100,00-	CLOSED
1100	0000001100	C	50,00-	50,00	100,00-	CLOSED
1100	0000001100	C	50,00-	50,00	100,00-	CLOSED
1100	0000001100	O	100,00	100,00	0,00	OPEN
1100	0000001100	O	100,00	100,00	0,00	OPEN
1100	0000001100	O	100,00	100,00	0,00	OPEN
1100	0000001100	O	300,00	300,00	0,00	OPEN
1100	0000001100	O	180,00	180,00	0,00	OPEN
1100	0000001100	O	100,00	100,00	0,00	OPEN
1100	0000001100	O	100,00	100,00	0,00	OPEN
1100	0000001100	O	10,00	10,00	0,00	OPEN

Expert Routine

Expert routines are used in special cases. In this example, a record is split into five records. A record from DataSource is split into five records and will be updated into cube as five records.

Create a DataSource and Cube and create a transformation between them.



Transformation Create

Transformation	RSDS ZEXPORT PC_FILE -> CUBE ZEXPORT_C
Source	ZEXPORT (ZEXPORT)
Target	Info Cube to display Expert Routine characterstics (ZEXPORT_C)
Version	New — Not saved
Active Version	Do(es) Not Exist — Edited Version

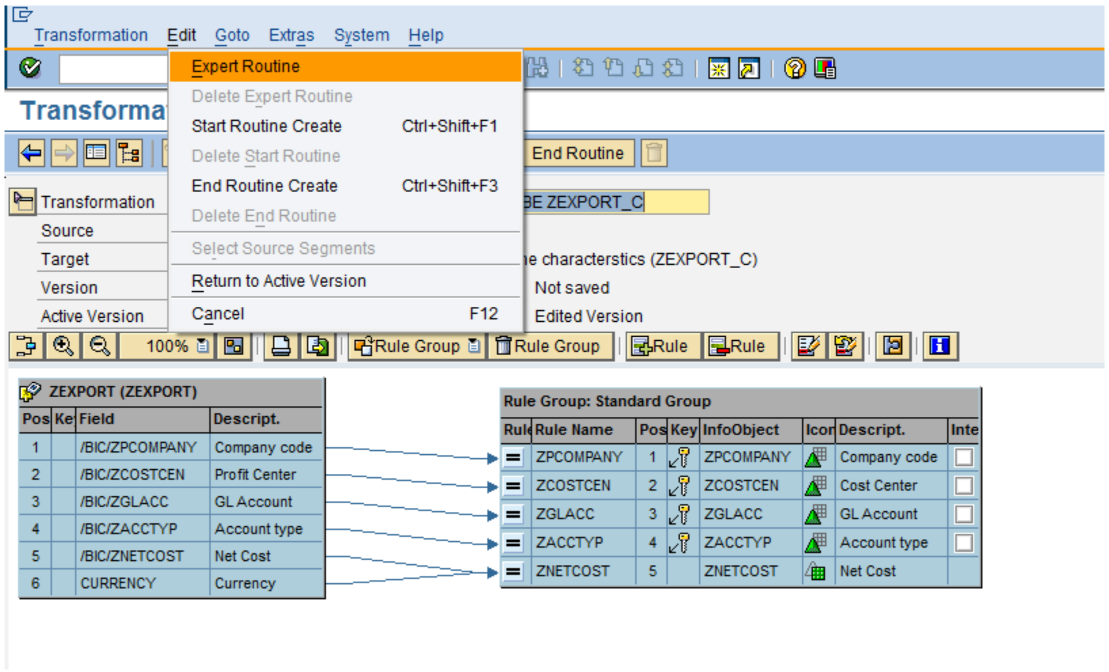
ZEXPORT (ZEXPORT)

Pos	Ke	Field	Descript.
1		/BIC/ZPCOMPANY	Company code
2		/BIC/ZCOSTCEN	Profit Center
3		/BIC/ZGLACC	GL Account
4		/BIC/ZACCTYP	Account type
5		/BIC/ZNETCOST	Net Cost
6		CURRENCY	Currency

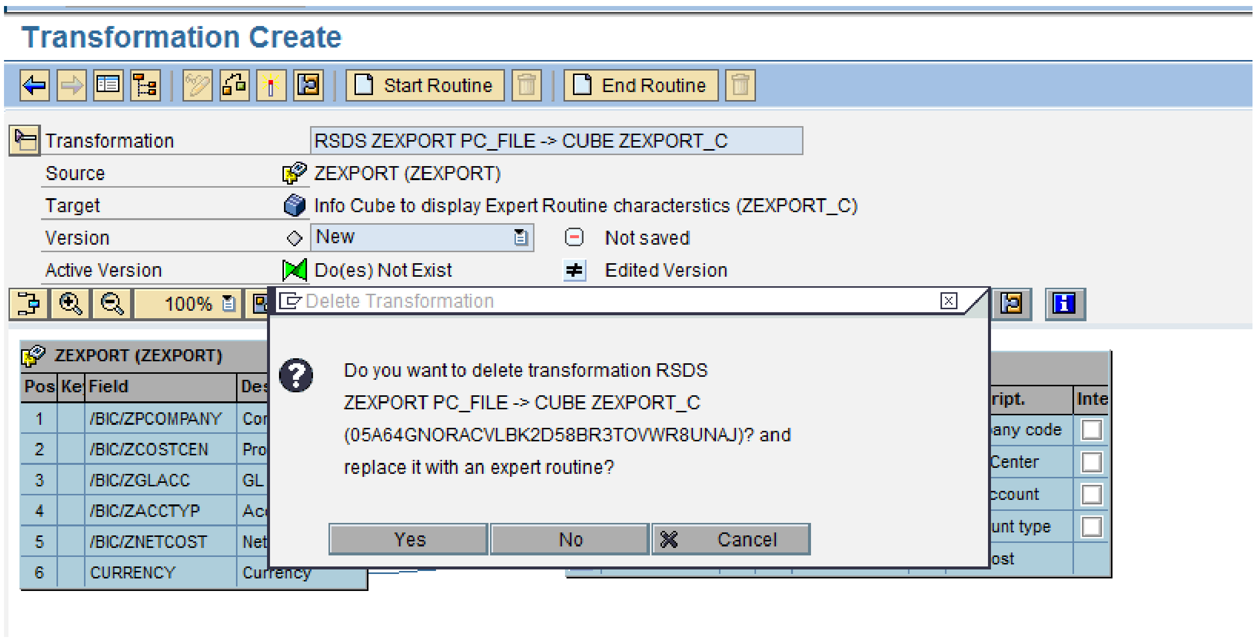
Rule Group: Standard Group

Rul	Rule Name	Pos	Key	InfoObject	Icon	Descript.	Inte
=	ZPCOMPANY	1		ZPCOMPANY		Company code	☐
=	ZCOSTCEN	2		ZCOSTCEN		Cost Center	☐
=	ZGLACC	3		ZGLACC		GL Account	☐
=	ZACCTYP	4		ZACCTYP		Account type	☐
=	ZNETCOST	5		ZNETCOST		Net Cost	

In the Menu Goto Edit->Expert Routine.



A pop up appears, whether the delete the transformation between source and target and replace it with Expert routine. Click ‘YES’



Transformation: RSDS ZEXPORT PC_FILE -> CUBE ZEXPORT_C
Source: ZEXPORT (ZEXPORT)
Target: Info Cube to display Expert Routine characterstics (ZEXPORT_C)
Version: New — Not saved
Active Version: Do(es) Not Exist — Edited Version

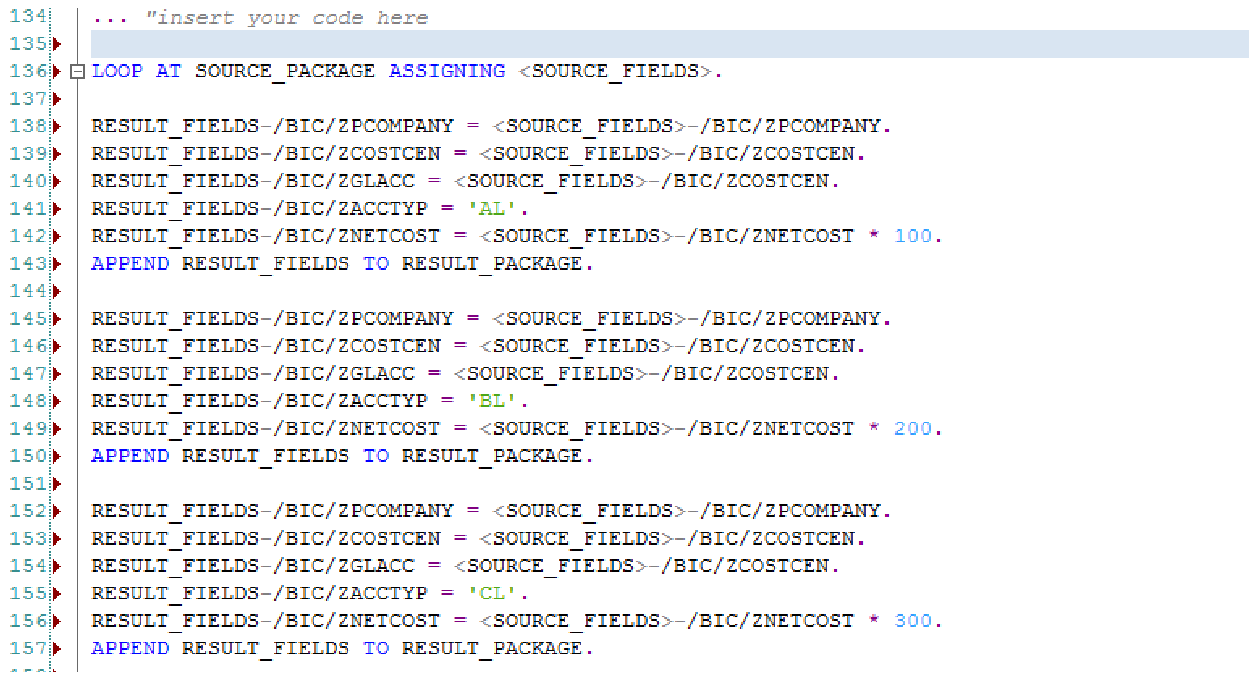
Delete Transformation

Do you want to delete transformation RSDS ZEXPORT PC_FILE -> CUBE ZEXPORT_C (05A64GNORACVLBK2D58BR3TOVWR8UNAJ)? and replace it with an expert routine?

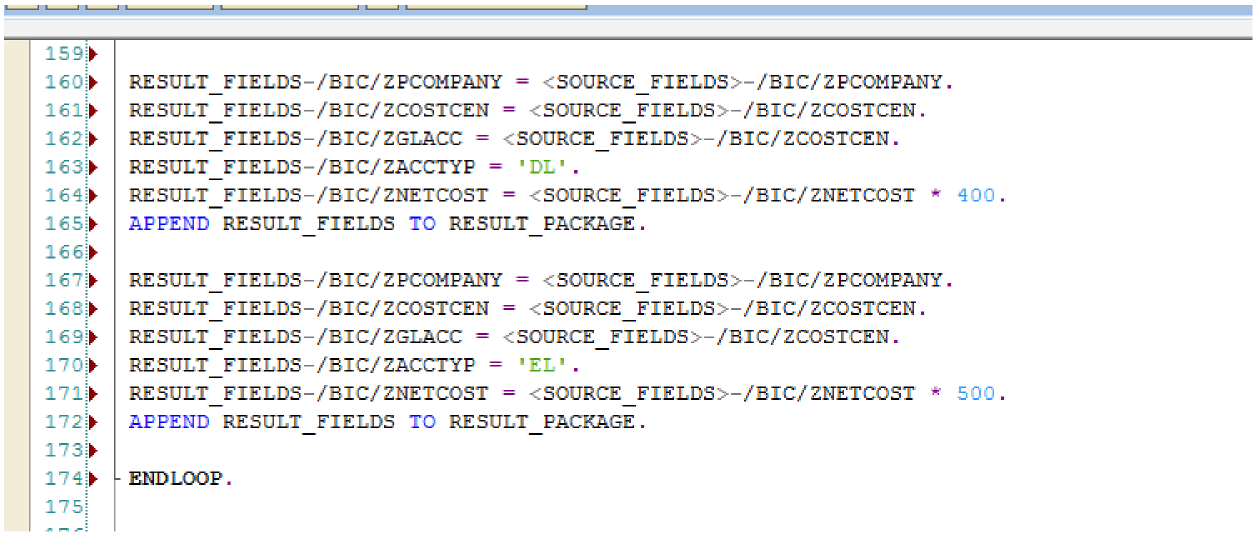
ZEXPORT (ZEXPORT)

Pos	Ke	Field	Des
1		/BIC/ZPCOMPANY	Co
2		/BIC/ZCOSTCEN	Pro
3		/BIC/ZGLACC	GL
4		/BIC/ZACCTYP	Ac
5		/BIC/ZNETCOST	Net
6		CURRENCY	Currency

Insert the required code. In this code, the account type (ZACCTYP) has been updated manually and according to the type, the cost has been calculated.

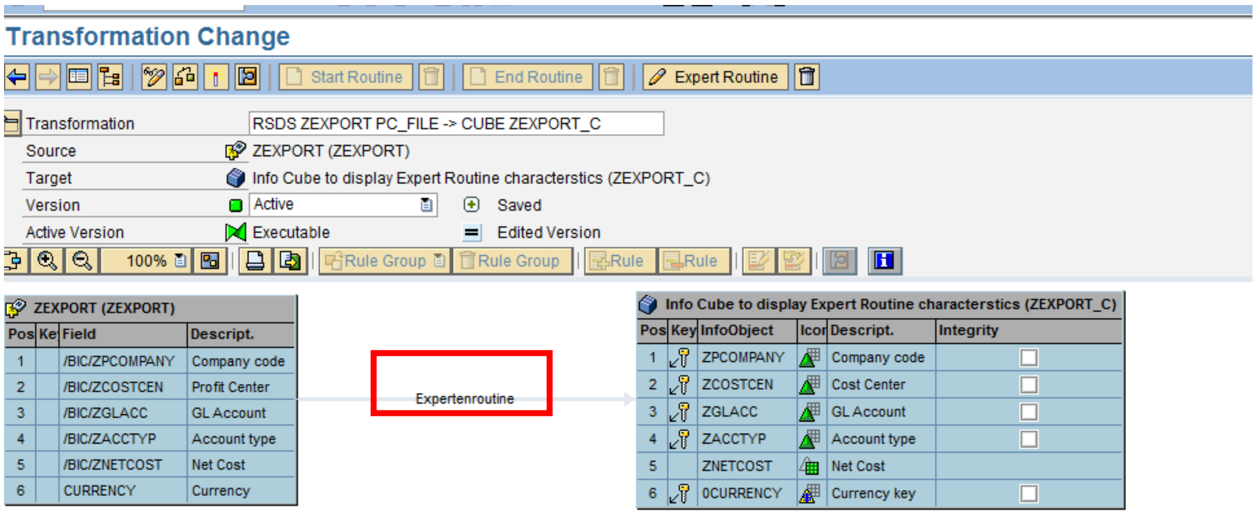


```
... "insert your code here

LOOP AT SOURCE_PACKAGE ASSIGNING <SOURCE_FIELDS>.

RESULT_FIELDS-/BIC/ZPCOMPANY = <SOURCE_FIELDS>-/BIC/ZPCOMPANY.
RESULT_FIELDS-/BIC/ZCOSTCEN = <SOURCE_FIELDS>-/BIC/ZCOSTCEN.
RESULT_FIELDS-/BIC/ZGLACC = <SOURCE_FIELDS>-/BIC/ZCOSTCEN.
RESULT_FIELDS-/BIC/ZACCTYP = 'AL'.
RESULT_FIELDS-/BIC/ZNETCOST = <SOURCE_FIELDS>-/BIC/ZNETCOST * 100.
APPEND RESULT_FIELDS TO RESULT_PACKAGE.

RESULT_FIELDS-/BIC/ZPCOMPANY = <SOURCE_FIELDS>-/BIC/ZPCOMPANY.
RESULT_FIELDS-/BIC/ZCOSTCEN = <SOURCE_FIELDS>-/BIC/ZCOSTCEN.
RESULT_FIELDS-/BIC/ZGLACC = <SOURCE_FIELDS>-/BIC/ZCOSTCEN.
RESULT_FIELDS-/BIC/ZACCTYP = 'BL'.
RESULT_FIELDS-/BIC/ZNETCOST = <SOURCE_FIELDS>-/BIC/ZNETCOST * 200.
APPEND RESULT_FIELDS TO RESULT_PACKAGE.

RESULT_FIELDS-/BIC/ZPCOMPANY = <SOURCE_FIELDS>-/BIC/ZPCOMPANY.
RESULT_FIELDS-/BIC/ZCOSTCEN = <SOURCE_FIELDS>-/BIC/ZCOSTCEN.
RESULT_FIELDS-/BIC/ZGLACC = <SOURCE_FIELDS>-/BIC/ZCOSTCEN.
RESULT_FIELDS-/BIC/ZACCTYP = 'CL'.
RESULT_FIELDS-/BIC/ZNETCOST = <SOURCE_FIELDS>-/BIC/ZNETCOST * 300.
APPEND RESULT_FIELDS TO RESULT_PACKAGE.
```



```

    RESULT_FIELDS-/BIC/ZPCOMPANY = <SOURCE_FIELDS>-/BIC/ZPCOMPANY.
    RESULT_FIELDS-/BIC/ZCOSTCEN = <SOURCE_FIELDS>-/BIC/ZCOSTCEN.
    RESULT_FIELDS-/BIC/ZGLACC = <SOURCE_FIELDS>-/BIC/ZCOSTCEN.
    RESULT_FIELDS-/BIC/ZACCTYP = 'DL'.
    RESULT_FIELDS-/BIC/ZNETCOST = <SOURCE_FIELDS>-/BIC/ZNETCOST * 400.
    APPEND RESULT_FIELDS TO RESULT_PACKAGE.

    RESULT_FIELDS-/BIC/ZPCOMPANY = <SOURCE_FIELDS>-/BIC/ZPCOMPANY.
    RESULT_FIELDS-/BIC/ZCOSTCEN = <SOURCE_FIELDS>-/BIC/ZCOSTCEN.
    RESULT_FIELDS-/BIC/ZGLACC = <SOURCE_FIELDS>-/BIC/ZCOSTCEN.
    RESULT_FIELDS-/BIC/ZACCTYP = 'EL'.
    RESULT_FIELDS-/BIC/ZNETCOST = <SOURCE_FIELDS>-/BIC/ZNETCOST * 500.
    APPEND RESULT_FIELDS TO RESULT_PACKAGE.

  ENDLOOP.

```

Save and activate the routine.



Transformation Change

Transformation	RSDS ZEXPORT PC_FILE -> CUBE ZEXPORT_C
Source	ZEXPORT (ZEXPORT)
Target	Info Cube to display Expert Routine characterstics (ZEXPORT_C)
Version	Active — Saved
Active Version	Executable — Edited Version

ZEXPORT (ZEXPORT)

Pos	Ke	Field	Descript.
1		/BIC/ZPCOMPANY	Company code
2		/BIC/ZCOSTCEN	Profit Center
3		/BIC/ZGLACC	GL Account
4		/BIC/ZACCTYP	Account type
5		/BIC/ZNETCOST	Net Cost
6		CURRENCY	Currency

Info Cube to display Expert Routine characterstics (ZEXPORT_C)

Pos	Key	InfoObject	Icon	Descript.	Integrity
1	🔑	ZPCOMPANY		Company code	☐
2	🔑	ZCOSTCEN		Cost Center	☐
3	🔑	ZGLACC		GL Account	☐
4	🔑	ZACCTYP		Account type	☐
5		ZNETCOST		Net Cost	
6	🔑	0CURRENCY		Currency key	☐

There will be a line connecting the Source and Target saying Expert Routine.

BI Variables

The customer exit is designed as an enhancement to configure with customer-specific logic.

For example if the requirement is as such, there should be only certain values that are to be displayed while selection of values, then we can create a Variable and use it in the quey.

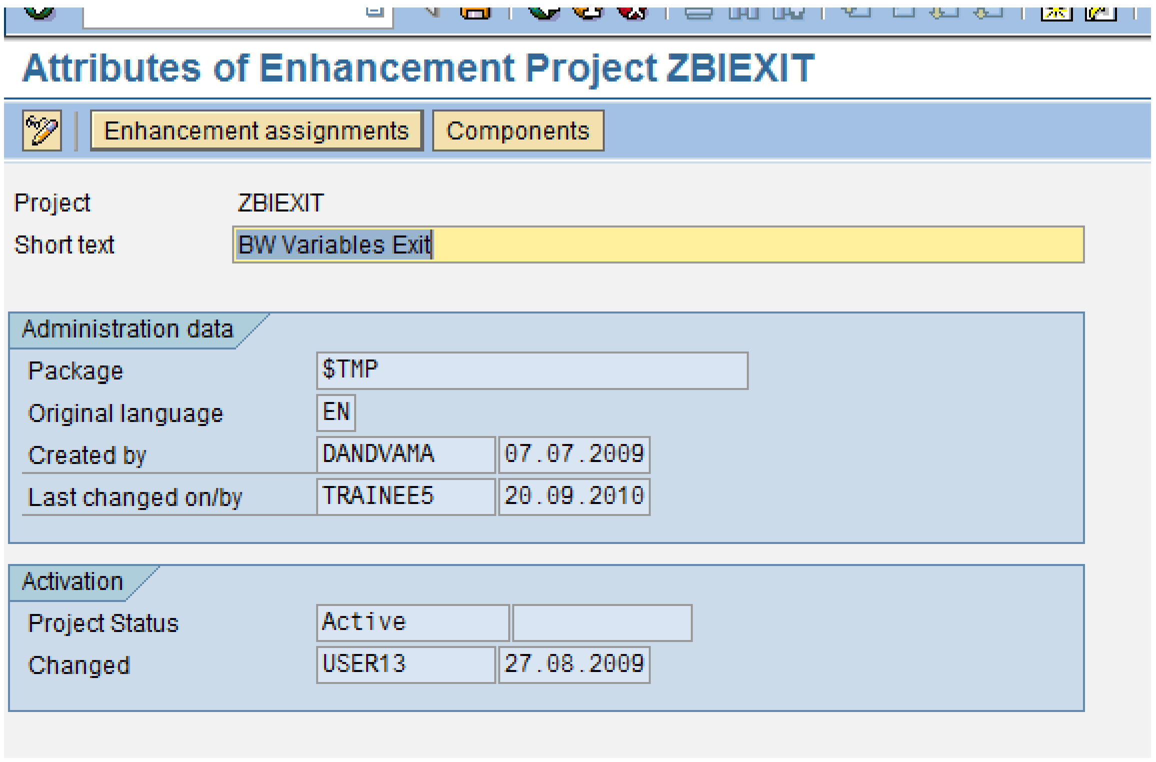
For a Customer Exit variable to be created, we need go to transaction code CMOD, create a new project and select SAP enhancement RSR00001 and assign it to the enhancement project.

Example

In this example, there is a variable which is created on posting date and which shows current day, in the input.

Steps for creating the variable:

Go to CMOD transaction code and create one Project.



Attributes of Enhancement Project ZBIEXIT

Project	ZBIEXIT	
Short text	BW Variables Exit	

Administration data

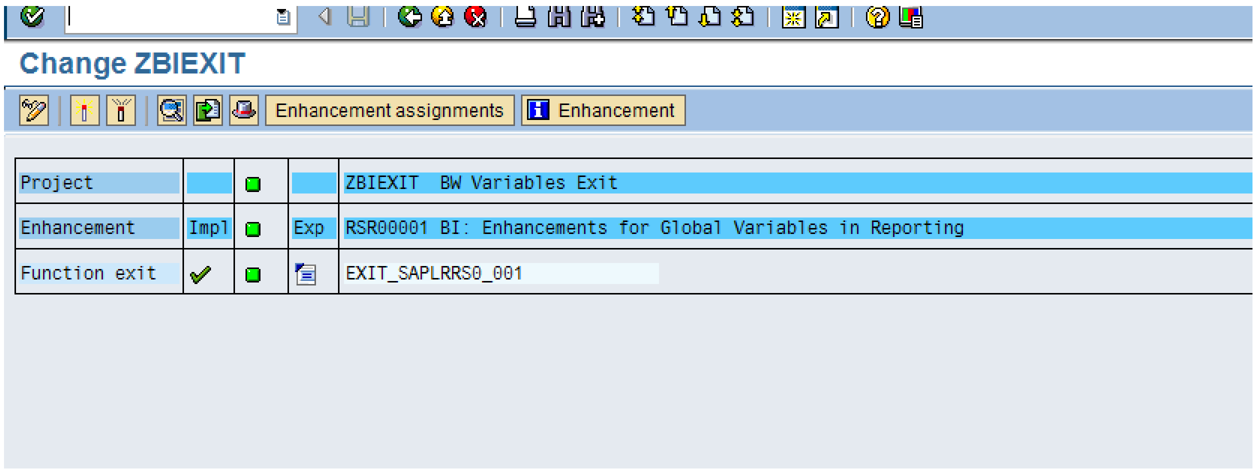
Package	$TMP	
Original language	EN	
Created by	DANDVAMA	07.07.2009
Last changed on/by	TRAINEE5	20.09.2010

Activation

Project Status	Active	
Changed	USER13	27.08.2009

Assign the Enhancement component as ‘RSR00001’.

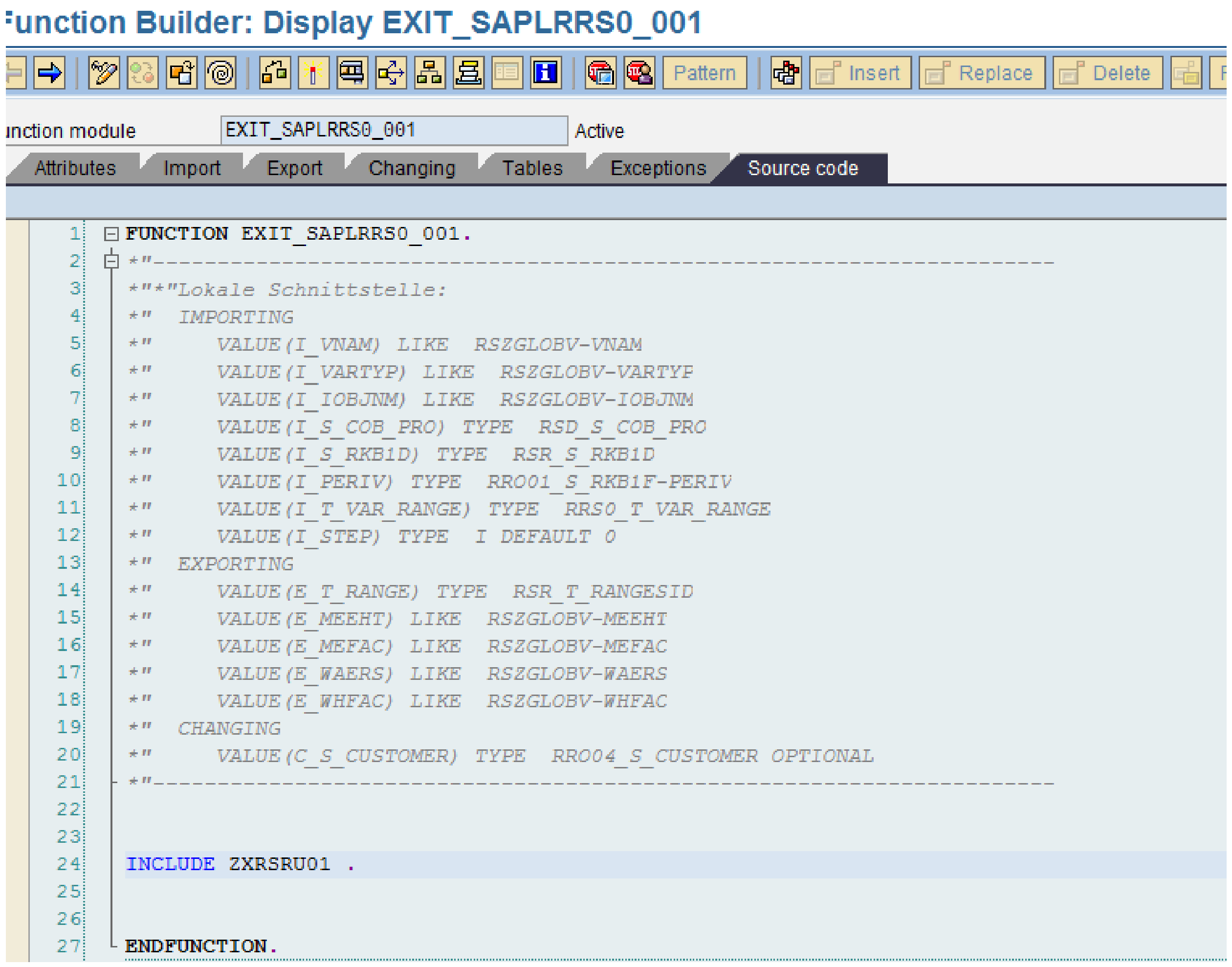
In the Components Screen, there is an Exit available and the name of the Exit is ‘EXIT_SAPLRRS0_001’, and we can write the code as per requirement.



Change ZBIEXIT

Project		🟩		ZBIEXIT BW Variables Exit
Enhancement	Impl	🟩	Exp	RSR00001 BI: Enhancements for Global Variables in Reporting
Function exit	✓	🟩		EXIT_SAPLRRS0_001

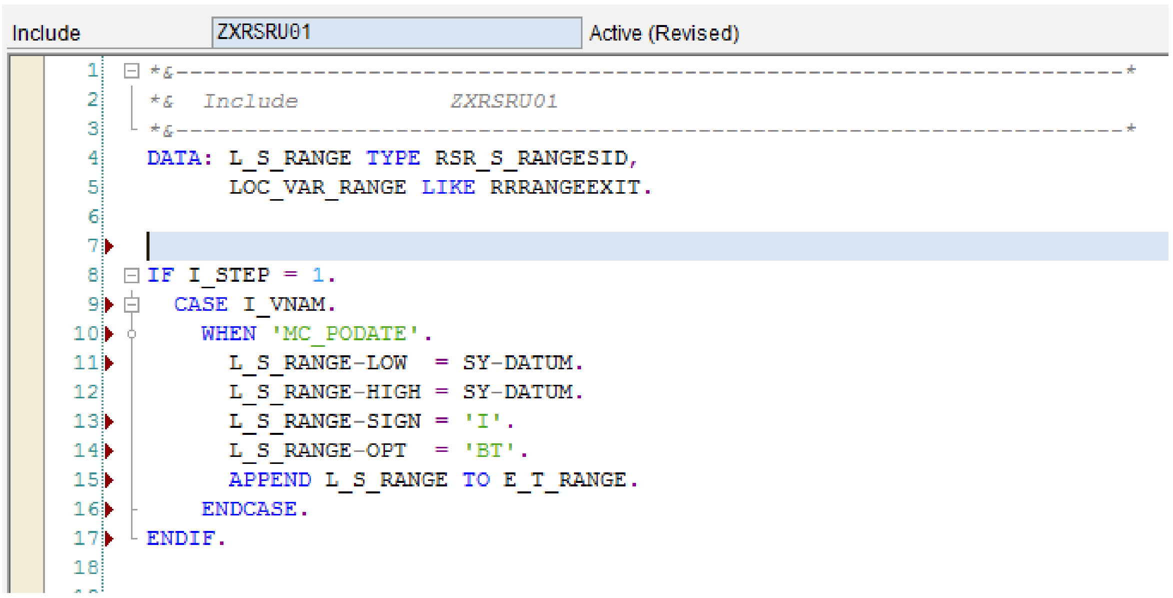
Once we click on the exit, we can see the below screen.



EXIT_SAPLRRS0_001 Active

```
FUNCTION EXIT_SAPLRRS0_001.
*"----------------------------------------------------------------------
*"*"Lokale Schnittstelle:
*"  IMPORTING
*"     VALUE(I_VNAM) LIKE  RSZGLOBV-VNAM
*"     VALUE(I_VARTYP) LIKE  RSZGLOBV-VARTYP
*"     VALUE(I_IOBJNM) LIKE  RSZGLOBV-IOBJNM
*"     VALUE(I_S_COB_PRO) TYPE  RSD_S_COB_PRO
*"     VALUE(I_S_RKB1D) TYPE  RSR_S_RKB1D
*"     VALUE(I_PERIV) TYPE  RRO01_S_RKB1F-PERIV
*"     VALUE(I_T_VAR_RANGE) TYPE  RRS0_T_VAR_RANGE
*"     VALUE(I_STEP) TYPE  I DEFAULT 0
*"  EXPORTING
*"     VALUE(E_T_RANGE) TYPE  RSR_T_RANGESID
*"     VALUE(E_MEEHT) LIKE  RSZGLOBV-MEEHT
*"     VALUE(E_MEFAC) LIKE  RSZGLOBV-MEFAC
*"     VALUE(E_WAERS) LIKE  RSZGLOBV-WAERS
*"     VALUE(E_WHFAC) LIKE  RSZGLOBV-WHFAC
*"  CHANGING
*"     VALUE(C_S_CUSTOMER) TYPE  RRO04_S_CUSTOMER OPTIONAL
*"----------------------------------------------------------------------


INCLUDE ZXRSRU01 .


ENDFUNCTION.
```

Double click on the Include, we can see ABAP Editor. Here we can write the logic. In this example, we area populating the System date (SY-DATUM) into posting date. So when the query executes, the Input will be by default Today's date instead of Posting Date.

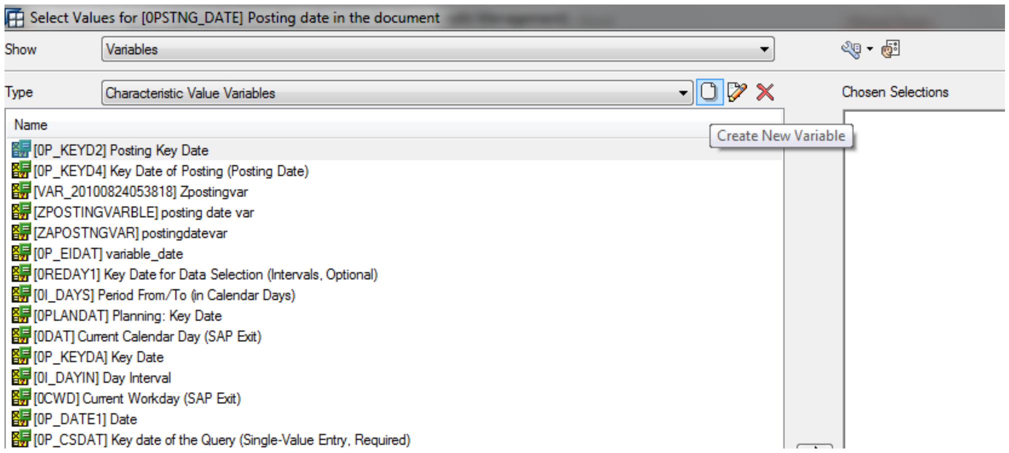


Include ZXRSRU01 Active (Revised)

```
*&---------------------------------------------------------------------*
*&  Include           ZXRSRU01
*&---------------------------------------------------------------------*
DATA: L_S_RANGE TYPE RSR_S_RANGESID,
      LOC_VAR_RANGE LIKE RRRANGEEXIT.


IF I_STEP = 1.
  CASE I_VNAM.
    WHEN 'MC_PODATE'.
      L_S_RANGE-LOW   = SY-DATUM.
      L_S_RANGE-HIGH  = SY-DATUM.
      L_S_RANGE-SIGN  = 'I'.
      L_S_RANGE-OPT   = 'BT'.
      APPEND L_S_RANGE TO E_T_RANGE.
    ENDCASE.
ENDIF.
```

Query:

In the query, for posting date create a variable.



Select Values for [0PSTNG_DATE] Posting date in the document

Show: Variables

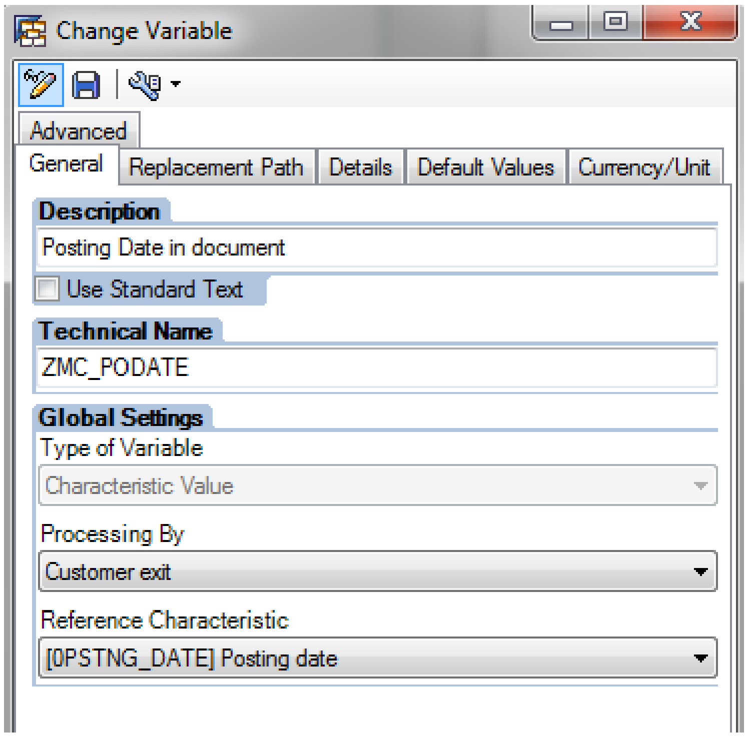
Type: Characteristic Value Variables

Create New Variable

Chosen Selections

Name
- [0P_KEYD2] Posting Key Date
- [0P_KEYD4] Key Date of Posting (Posting Date)
- [VAR_20100824053818] Zpostingvar
- [ZPOSTINGVARBLE] posting date var
- [ZAPOSTNGVAR] postingdatevar
- [0P_EIDAT] variable_date
- [0REDAY1] Key Date for Data Selection (Intervals, Optional)
- [0I_DAYS] Period From/To (in Calendar Days)
- [0PLANDAT] Planning: Key Date
- [0DAT] Current Calendar Day (SAP Exit)
- [0P_KEYDA] Key Date
- [0I_DAYIN] Day Interval
- [0CWD] Current Workday (SAP Exit)
- [0P_DATE1] Date
- [0P_CSDAT] Key date of the Query (Single-Value Entry, Required)

Properties of the Variable



Change Variable

Advanced | General | Replacement Path | Details | Default Values | Currency/Unit

Description: Posting Date in document

Use Standard Text: ☐

Technical Name: ZMC_PODATE

Global Settings

Type of Variable: Characteristic Value

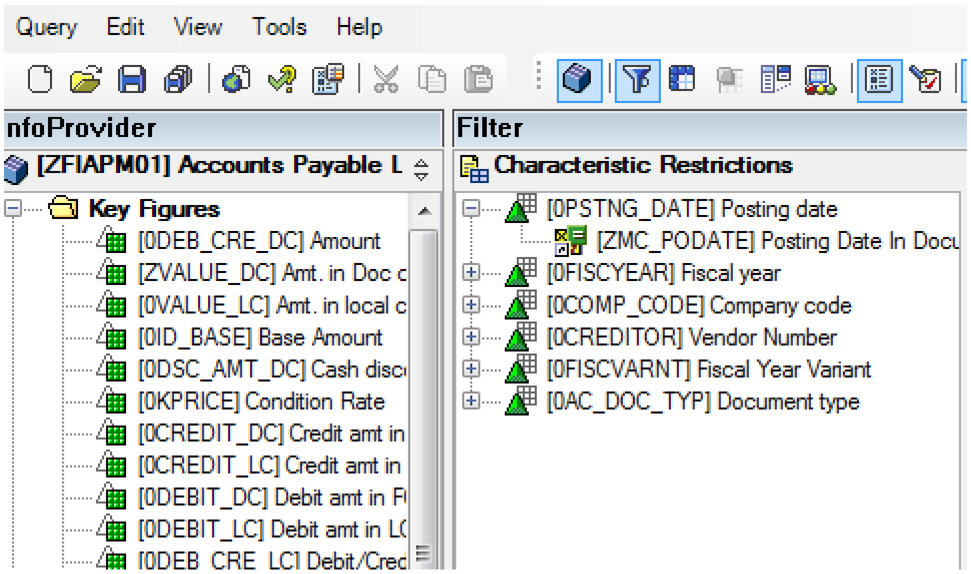
Processing By: Customer exit

Reference Characteristic: [0PSTNG_DATE] Posting date

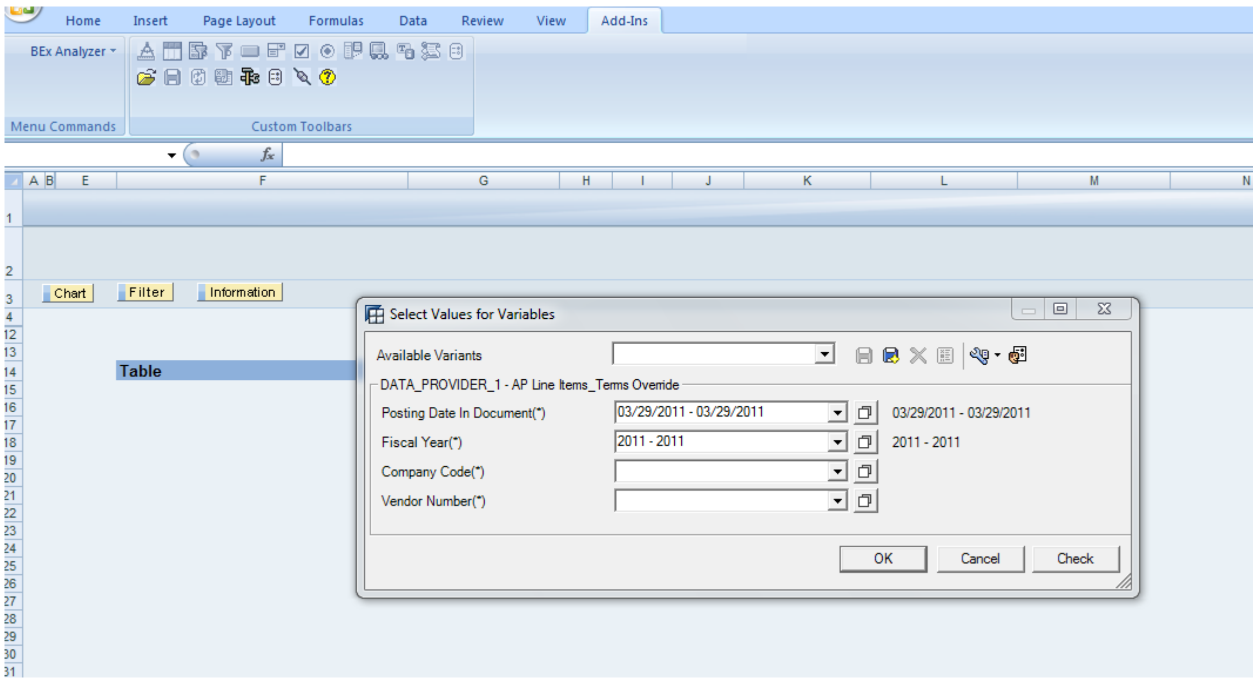
The Processing type should be mentioned as ‘Customer Exit’.

Save the Variable once done.

Use the Variable for Posting Date in Query.


Query
Edit
View
Tools
Help
nfoProvider
Filter
[ZFIAPM01] Accounts Payable L
Characteristic Restrictions
Key Figures
[0DEB_CRE_DC] Amount
[ZVALUE_DC] Amt. in Doc c
[0VALUE_LC] Amt. in local c
[0ID_BASE] Base Amount
[0DSC_AMT_DC] Cash disc
[0KPRICE] Condition Rate
[0CREDIT_DC] Credit amt in
[0CREDIT_LC] Credit amt in
[0DEBIT_DC] Debit amt in F
[0DEBIT_LC] Debit amt in L
[0DEB_CRE_LC] Debit/Cred
[0PSTNG_DATE] Posting date
[ZMC_PODATE] Posting Date In Docu
[0FISCYEAR] Fiscal year
[0COMP_CODE] Company code
[0CREDITOR] Vendor Number
[0FISCVARNT] Fiscal Year Variant
[0AC_DOC_TYP] Document type

After Executing the Query, The variable screen is shown below.



DATA_PROVIDER_1 - AP Line Items_Terms Override

Posting Date In Document(*)	03/29/2011 - 03/29/2011	03/29/2011 - 03/29/2011
Fiscal Year(*)	2011 - 2011	2011 - 2011
Company Code(*)		
Vendor Number(*)		

Related Content

For more information, visit the EDW homepage.

Disclaimer and Liability Notice