**Example of a Start Routine**

*Цель* - You want to create a start routine that deletes all the records from a data package with debit and credit postings /проводки/ that are zero.

1. Create a transformation. The source of the transformation has the Total Debit Postings /UMSOL/ and Total Credit Postings /UMHAB/ fields. They are assigned to the InfoObjects Total Debit Postings /0DEBIT/ and Total Credit Postings /0CREDIT/.
2. You are in the BW modeling tools, editing your transformation on the General tab. Choose ***Start Routine  Create***. The routine editor opens.
3. Go to the local part of the routine. Enter the following code

METHOD ***start\_routine***.

FIELD-SYMBOLS:

<SOURCE\_FIELDS> TYPE \_ty\_s\_SC\_1.

…

*\* удалить все записи, где кредитные и дебетные проводки равны 0*

DELETE SOURCE\_PACKAGE where UMHAB = 0 and UMSOL = 0.

ENDMETHOD.

The DELETE statement is the only line you need to filter debit and credit postings without values from the data package.

* Save the class and go back to the transformation editor.
* Activate the transformation. The transformation has object status Active, but the version status is Revised.

**Example of Characteristic Routine**

*Цель* - You want to create a routine for the characteristic Debit/Credit Indicator /0FI\_DBCRIND/ in the target that assigns the value D to debit postings and the value C to credit postings.

1. You are in the BW modeling tools, editing your transformation on the Rules tab. Select the InfoObject Debit / Credit Indicator /0FI\_DBCRIND/ and the rule detailsare displayed in the Properties View.
2. You choose Add Fields and add the Total Debit Postings /UMSOL/ and Total Credit Postings /UMHAB/ fields so that they are available in the routine.
3. You choose Routine as the rule type. When you choose Routine  Edit the class editor opens.
4. You enter the following lines of code. They return either a D or a C as the result value.

METHOD ***compute\_0FI\_DBCRIND***.

DATA:

MONITOR\_REC TYPE rsmonitor.

*\* result value of the routine*

if SOURCE\_FIELDS-*umhab* ne 0 and SOURCE\_FIELDS-umsol eq 0.

*RESULT = 'D'*.

elseif SOURCE\_FIELDS-*umhab* eq 0 and SOURCE\_FIELDS-umsol ne 0.

RESULT = 'C'.

else.

monitor\_rec-msgid = 'ZMESSAGE'.

monitor\_rec-msgty = 'E'.

monitor\_rec-msgno = '001'.

monitor\_rec-msgv1 = 'ERROR, D/C Indicator'.

monitor\_rec-msgv2 = SOURCE\_FIELDS-umhab.

monitor\_rec-msgv3 = SOURCE\_FIELDS-umsol.

append monitor\_rec to monitor.

RAISE EXCEPTION TYPE CX\_RSROUT\_ABORT.

endif.

ENDMETHOD.

*The system checks whether the debit and credit postings contain values*

* + If the debit posting has values that are not equal to zero and the credit posting is equal to zero, the system assigns the value D.
  + If the credit posting has values that are not equal to zero and the debit posting is equal to zero, the system assigns the value C.
  + If both the debit and credit postings contain values, the system outputs an error in the monitor and terminates the load process.

1. Save and activate the class, and close the class editor.
2. You save the transformation.

**Example of an End Routine**

*Цель* - You want to create an end routine to fill the additional InfoObject Plan/Actual Indicator /ZPLACTUAL/. You also want the routine to read the Value Type field. If the value is 10 /actual/, value A is written to the Plan/Actual Indicator InfoObject; if the value is 20 /plan/, value P is written to the Plan/Actual Indicator InfoObject.

1. You are in the BW modeling tools, editing your transformation on the General tab. Choose End Routine  Create ABAP. The routine editor opens.
2. You enter the following lines of code.

METHOD ***end\_routine***.

FIELD-SYMBOLS:

<RESULT\_FIELDS> TYPE \_ty\_s\_TG\_1.

loop at *RESULT\_PACKAGE* assigning <RESULT\_FIELDS>

where vtype eq '010' or vtype eq '020'.

case <RESULT\_FIELDS>-vtype.

when '010'.

<RESULT\_FIELDS>-/bic/zplactual = 'A'. "Actual

when '020'.

<RESULT\_FIELDS>-/bic/zplactual = 'P'. "Plan

endcase.

endloop.

ENDMETHOD.

The code loops through the result\_package searching for values that have the value type 10 or 20. For these values, the appropriate value is passed on to InfoObject Plan/Actual Indicator /ZPLACTUAL/.

1. Save the class and go back to the transformation editor.
2. Activate the transformation. The transformation has object status Active, but the version status is Revised.

# Writing quick lookup transformation code for SAP BW.

<https://www.acorel.nl/2012/04/writing-quick-lookup-transformation-code-for-sap-bw/>

### *Start routine*

[http://3.bp.blogspot.com/-oX-nMpOS9VQ/T5a72-kBclI/AAAAAAAAAK4/CnHR4OQ6DUw/s200/24-4-2012+16-38-01.jpg](http://3.bp.blogspot.com/-oX-nMpOS9VQ/T5a72-kBclI/AAAAAAAAAK4/CnHR4OQ6DUw/s1600/24-4-2012+16-38-01.jpg)

TYPES:

BEGIN OF *ty\_netw*,

             l\_netw       TYPE  /BI0/OINETWORK,

l\_wbs\_elemt  TYPE  /BI0/OIWBS\_ELEMT,

END OF ty\_netw.

DATA:

wa\_netw    TYPE ty\_netw,

t\_netw     LIKE TABLE OF  wa\_netw.

*\* Filling Internal table for NETWORK*

SELECT NETWORK WBS\_ELEMT

FROM /BI0/PNETWORK

INTO TABLE t\_netw

WHERE OBJVERS = ‘A’.

SORT t\_netw BY l\_wbs\_elemt.

### *Transformation routine*

[http://4.bp.blogspot.com/-ZuOV3m8nfo4/T5a7qLBaSJI/AAAAAAAAAKw/VMockpvMcqg/s400/24-4-2012+16-29-06.jpg](http://4.bp.blogspot.com/-ZuOV3m8nfo4/T5a7qLBaSJI/AAAAAAAAAKw/VMockpvMcqg/s1600/24-4-2012+16-29-06.jpg)

IF NOT COMM\_STRUCTURE-WBS\_ELEMT is initial.

clear wa\_netw.

READ TABLE t\_netw INTO wa\_netw WITH KEY

l\_wbs\_elemt  = COMM\_STRUCTURE-WBS\_ELEMT

BINARY SEARCH.

IF sy-subrc EQ 0. RESULT = wa\_netw-l\_netw.

ELSE. RESULT = ”.

ENDIF.

ELSE. RESULT = ”.

ENDIF.