Steps to Debug Routines in BI Transformations

Applies to:

SAP NetWeaver BW 3.x. For more information, visit the Business Intelligence homepage.

Summary

This article explains the Debugging process for routines in Transformations. It helps us to trace the errors crept in Routines in the debug mode, so that it can be handled effectively.

Author: Rudra Pradeep Reddy Neelapu

Company: Mahindra Satyam

Created on: 31 March 2010

Author Bio

Working as a SAP Technical consultant with Mahindra Satyam. Skill set includes SAP Business Intelligence, ABAP and Business Objects.

Table of Contents

Introduction

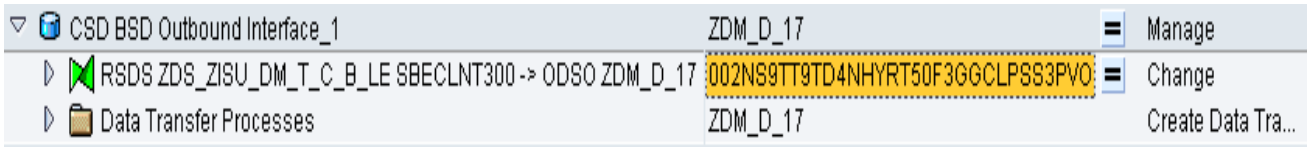
This article is mainly intended to provide an idea of how we can handle the debugging process for the routines we code in the transformations. It explains in steps the process to derive at a generated Program which contains our routine code. From here we can carry out with our normal debugging procedure.

Business Scenario

When we want to know about the errors crept in our routines, we need to debug the code we had written in transformation routines like start and update routines.

Procedure

Go with the Transformations which need to be debugged.



Let's have a look at the Start and Update Routine in the Transformations.

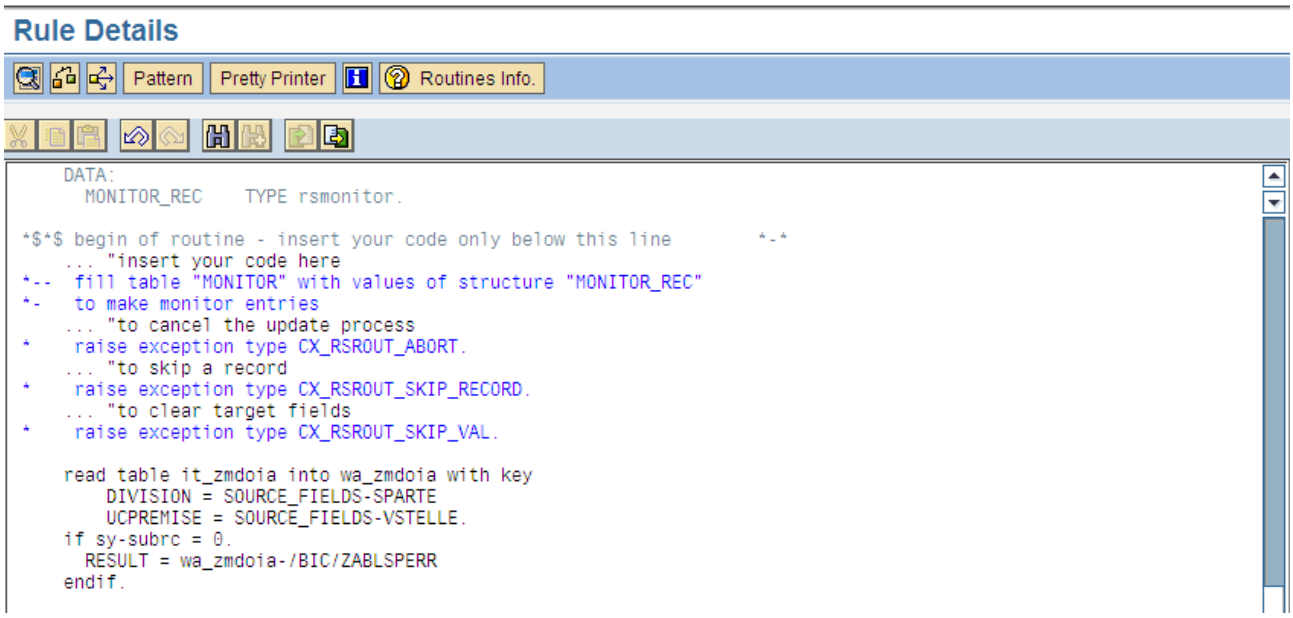
Start Routine of the Transformation:

Transformation Change

Pattern | Pretty Printer | Routines Info.

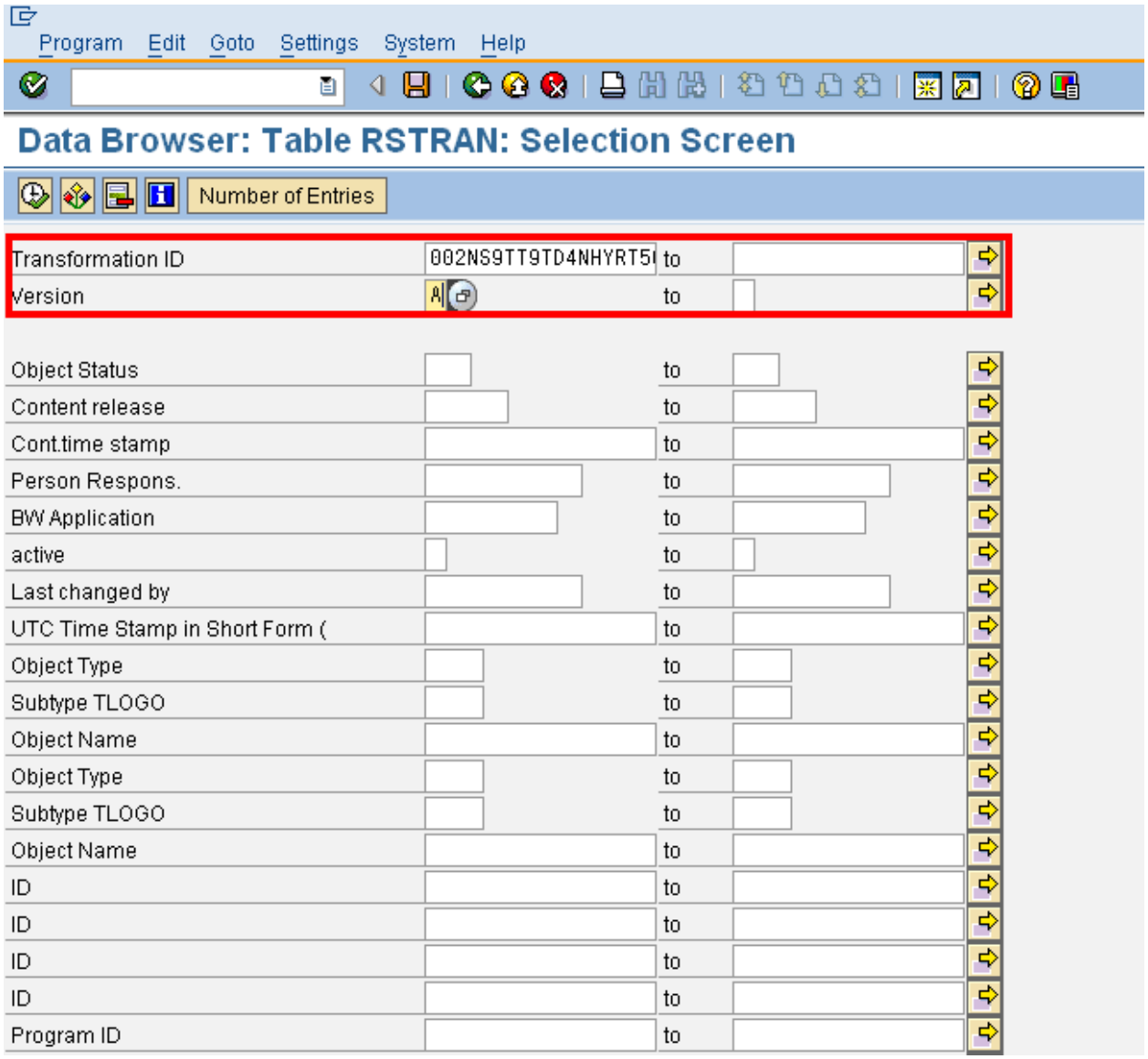
```
*$*$ begin of 2nd part global - insert your code only below this line  *
... "insert your code here


types: begin of ty_zmdoia,
      UCPREMISE type /BI0/OIUCPREMISE,
      DIVISION type /BI0/OIDIVISION,
      /BIC/ZABLSPERR type /BIC/OIZABLSPERR,
      end of ty_zmdoia,
      tt_zmdoia type standard table of ty_zmdoia.

data: it_zmdoia type tt_zmdoia,
      wa_zmdoia type ty_zmdoia.

Data : ZVAR type /BIC/OIZABLSPERR.

*$*$ end of 2nd part global - insert your code only before this line   *

*---------------------------------------------------------------------*
*       CLASS routine IMPLEMENTATION
*---------------------------------------------------------------------*
*
*---------------------------------------------------------------------*
CLASS lcl_transform IMPLEMENTATION.

*$*$ begin of routine - insert your code only below this line        *-*
    ... "insert your code here

    if not SOURCE_PACKAGE is initial.

    select  UCPREMISE DIVISION /BIC/ZABLSPERR
      from /BIC/AZMDOIA00 into table it_zmdoia
      for all entries in SOURCE_PACKAGE where
       ( ( UCPREMISE = SOURCE_PACKAGE-VSTELLE ) and
      ( DIVISION = SOURCE_PACKAGE-SPARTE ) ).
      .
      if sy-subrc = 0.
      sort it_zmdoia by UCPREMISE DIVISION.
      delete adjacent duplicates from it_zmdoia
      comparing UCPREMISE DIVISION.
      endif.
    endif.
```

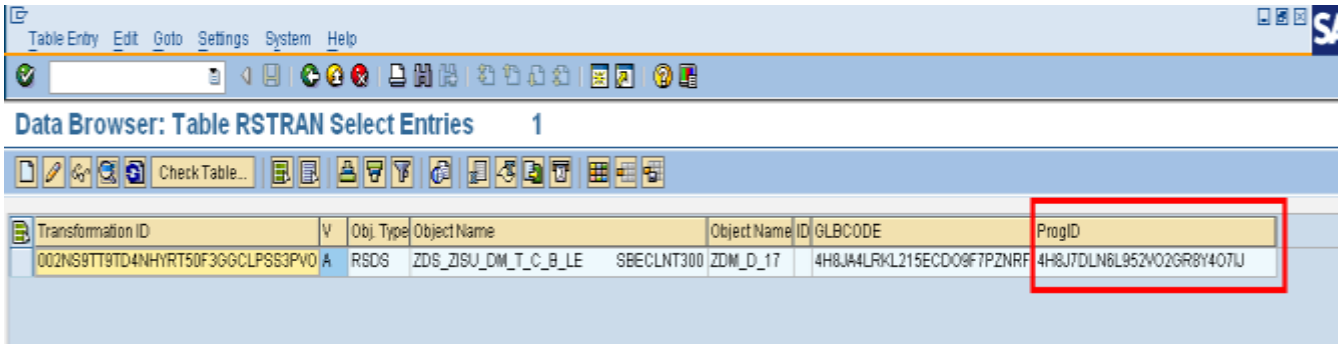
Update Routine:



To debug these routines we need to go to the generated program of the Transformations in two ways:

We can get the Program ID for the **Transformation ID (4H8JA4TG3JNQO0VTU3HK01MH7)** from table **RSTRAN.**

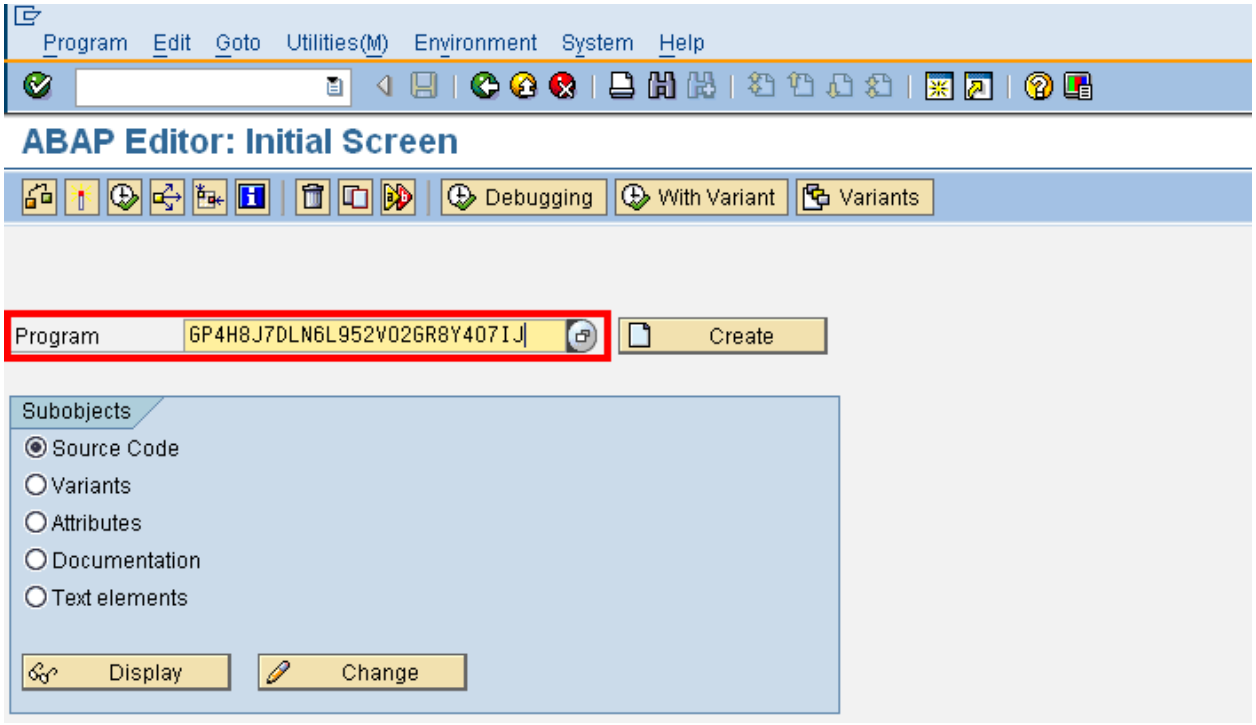




On Appending the ProgID (**4H8J7DLN6L952VO2GR8Y4O7IJ**) with **GP** we get the report program for the Transformations.

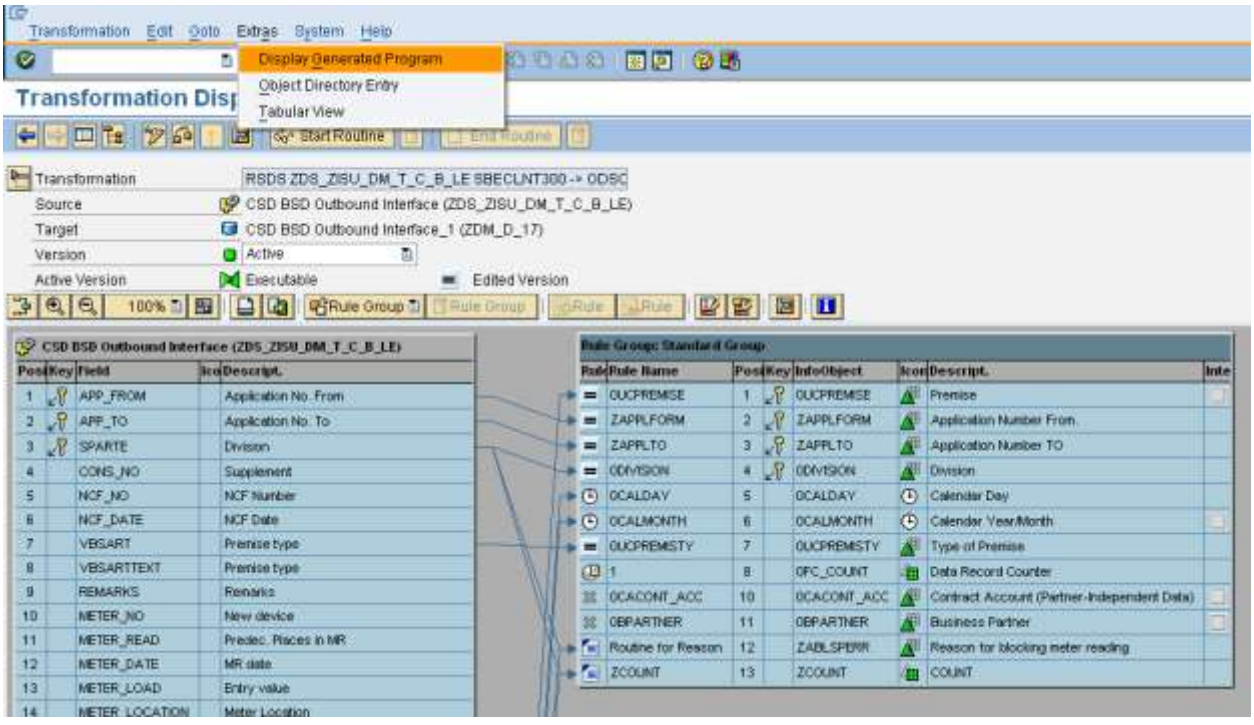
Go to SE38 and give the Program Name (**GP4H8J7DLN6L952VO2GR8Y4O7IJ**).

Generated Program

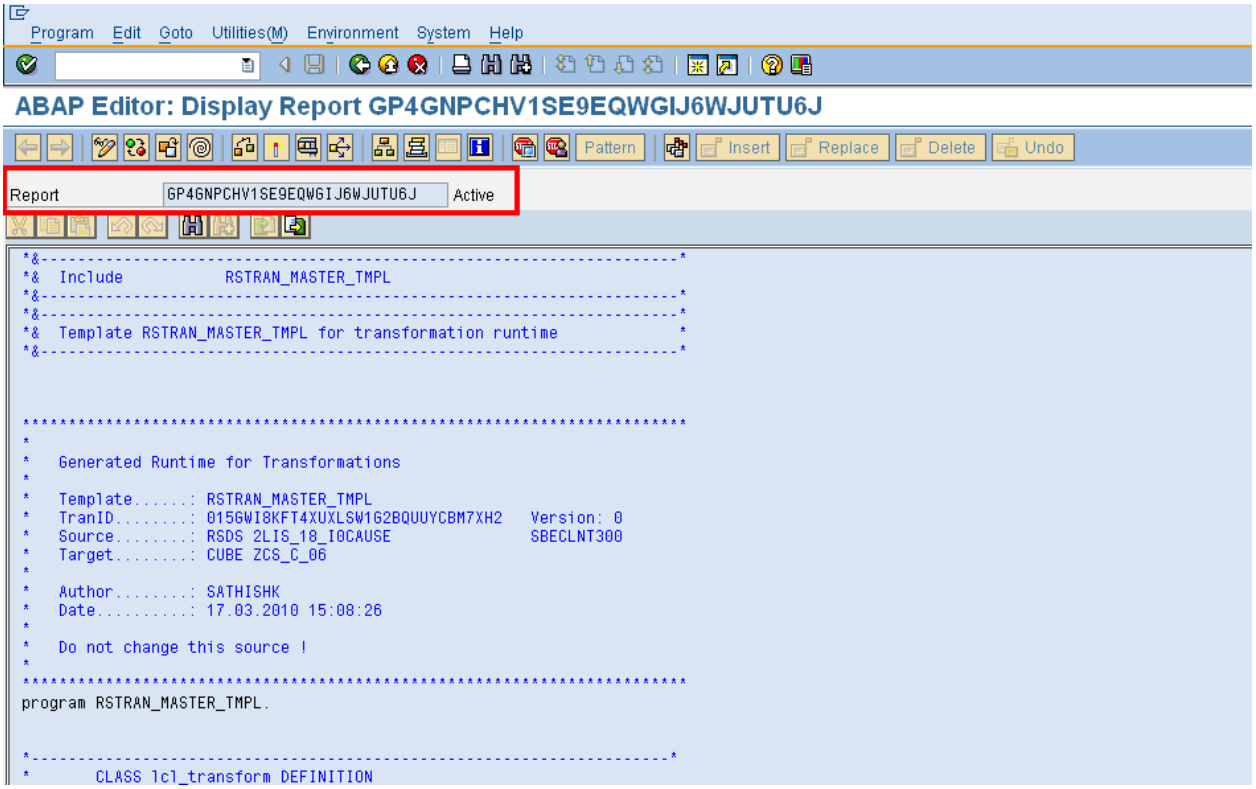


On going with display we will be to the ABAP editor of the Report Program. The other way of getting to this program is from the Transformations itself.

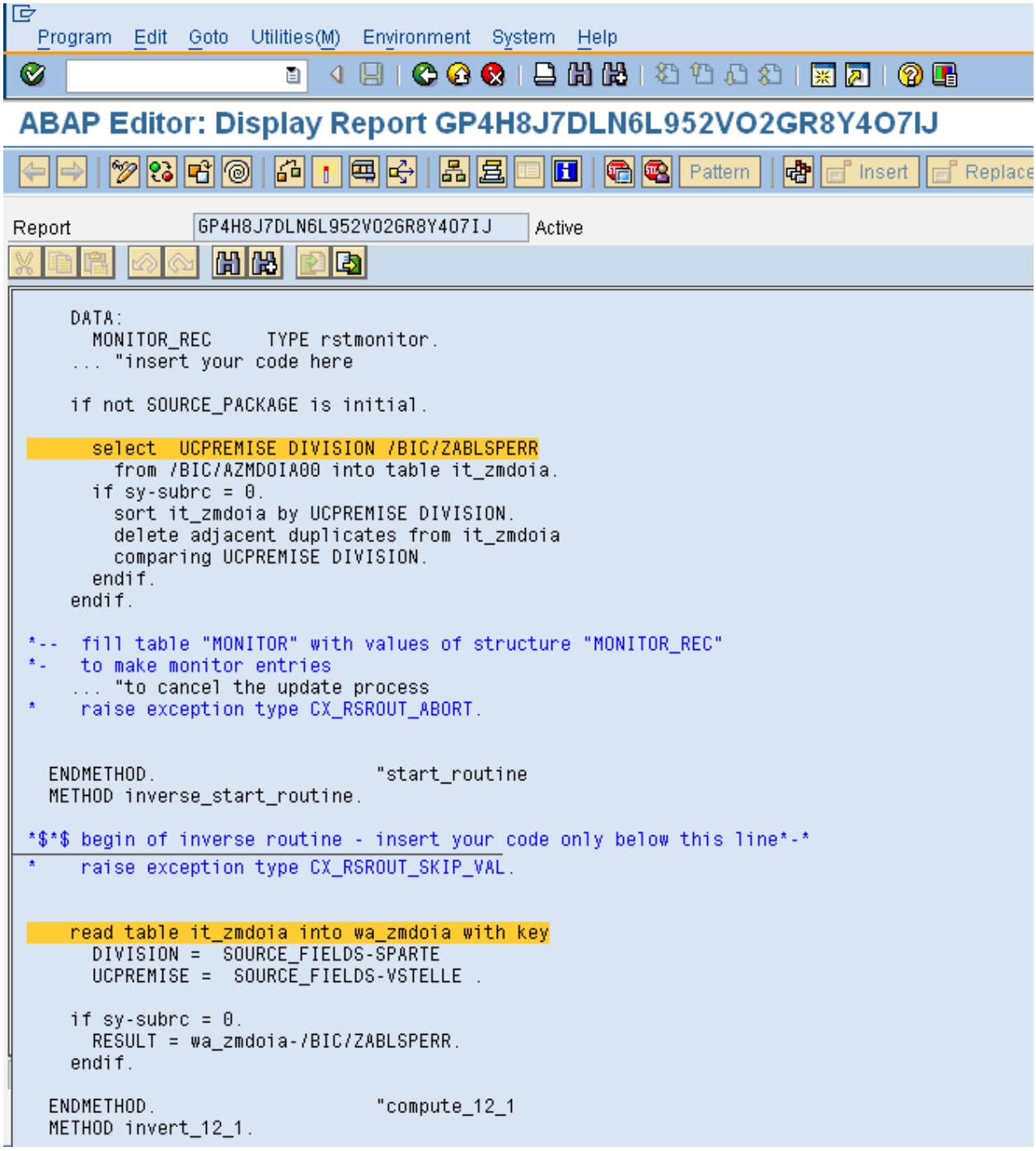
Go to Extras →Display Generated Program. It directly takes us to the ABAP editor of the Report Program.



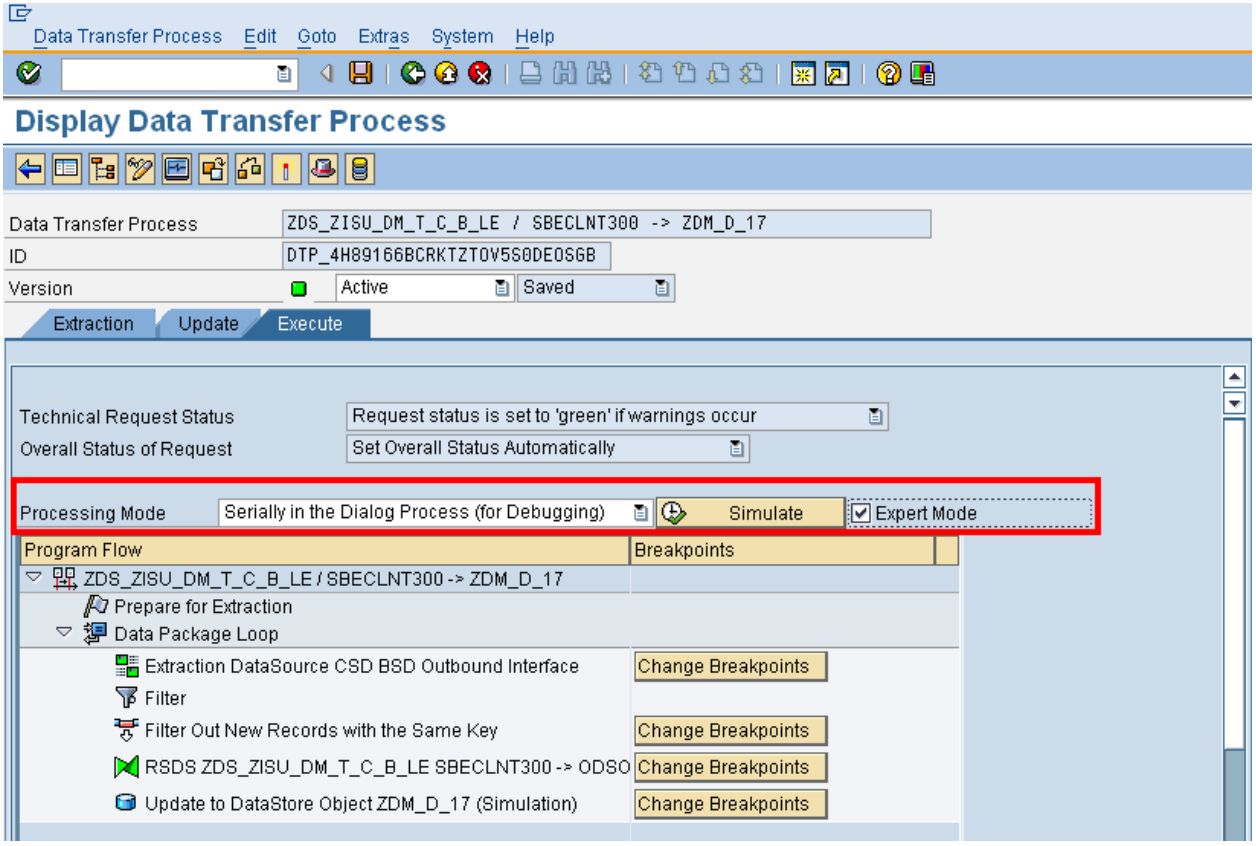
Below is the ABAP editor screen of the Report Program:



Set a break point at the code you want to debug, i.e. at your selections at start routine and at your update routine.



We can exit the screen and can debug it by running the DTP in Debugging Mode.



Select the Processing Mode Serially in the Dialog process (for Debugging) and go with simulate with Expert Mode. If you set this indicator, you can debug a DTP request in expert mode. The system does not execute the simulation request directly; you can edit the simulation request interactively before it is processed.

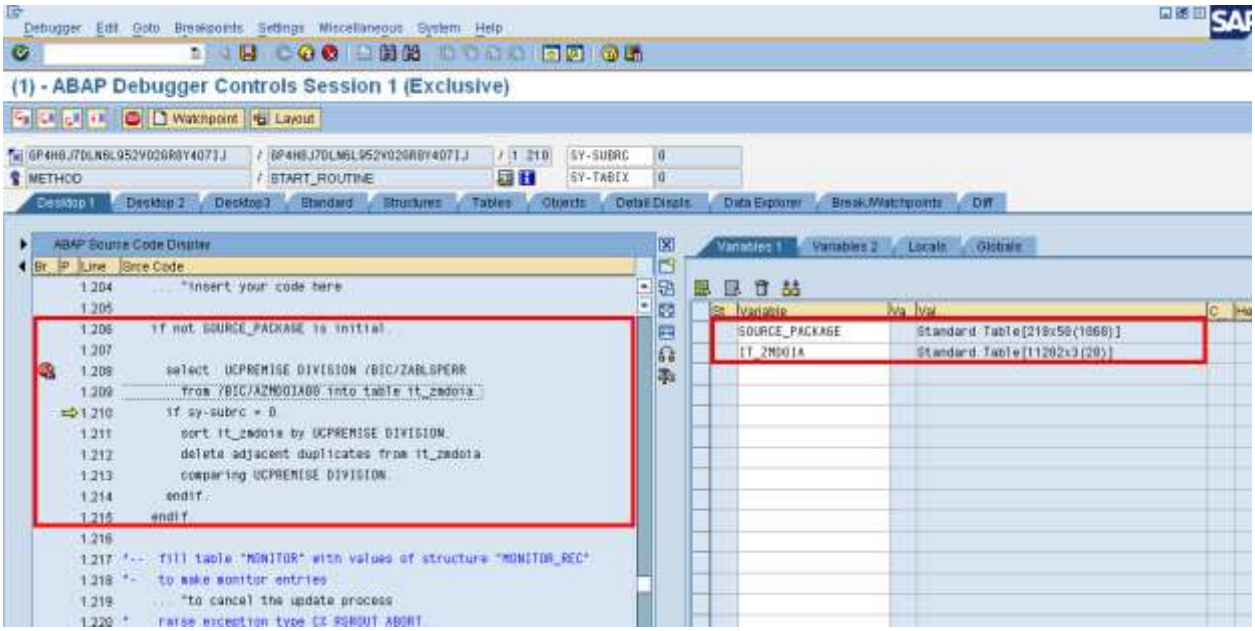
You can change the following properties:

- Selections
- Setting for temporary storage
- Breakpoints

We can use expert mode in the following cases:

- If the amount of data in the source is very large and the runtime of a simulation request for the entire selection would be considerable
- If you want to generate the temporary storage during request processing but do not want to change the DTP settings.
- This is useful in productive systems where you cannot change DTPs.
- If you want to test a particular record

On going with Simulate we are with the ABAP Debugger Session. Now we can go with normal debugging process.



Here we can find the entries in the SOURCE_PACKAGE and the entries fetched into the internal table IT_ZMDOIA in the start routine.

Related Content

Taken most of the inputs from **sap help** in preparing this Article.

http://forums.sdn.sap.com/click.jspa?searchID=41376436&messageID=8664110

http://forums.sdn.sap.com/click.jspa?searchID=41376436&messageID=8646524

For more information, visit the Business Intelligence homepage.

Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.