SAP BI Generic Extraction Using a Function Module

Applies to:
SAP BI

Summary
This article demonstrates a step-by-step process for doing generic extraction from R3 into BI using a Function Module.

Author(s): P Renjith Kumar.
Company: Pricol Technologies
Created on: 14 February 2007

Author Bio
P. Renjith Kumar is SAP BI Consultant with PRICOL Technologies. He has extensive cross-functional experience and has been with end-to-end SAP implementations as an ABAP Consultant.
Table of Contents

R3 Side............................................................................................................................... 3
  Code............................................................................................................................. 8
  Data Source Creation.............................................................................................. 10
  Checking Extraction................................................................................................. 11
  BW Side.................................................................................................................... 12
Disclaimer and Liability Notice...................................................................................... 17
R3 Side

1. Create a structure with the necessary fields. The Currency and quantity fields must have reference fields also.

   **NETWR** is a currency field so it must have reference field that is in the base table. Click Currency / Quan and give that field name.

2. In transaction SE80, Select Function group, name RSAX. Right click and Copy:
Select only the needed function module:
The Function Module in SE37 will be like this, we have to do light modification
No change in Import Tab.

Export, Changing No values.
In TABLES tab, give the structure name.

In Exceptions No data. In Source code Tab. Write this Code

The Code that we need to write is Placed Inside
FUNCTION ZRSAX_BIW_GET_DATA_SIMPLE4_2.

**Local interface:**
* IMPORTING
  VALUE(I_REQUNR) TYPE SRSC_S_IF_SIMPLE-REQUNR
  VALUE(I_DSOURCE) TYPE SRSC_S_IF_SIMPLE-DSOURCE OPTIONAL
  VALUE(I_MAXSIZE) TYPE SRSC_S_IF_SIMPLE-MAXSIZE OPTIONAL
  VALUE(I_INITFLAG) TYPE SRSC_S_IF_SIMPLE-INITFLAG OPTIONAL
  VALUE(I_READ_ONLY) TYPE SRSC_S_IF_SIMPLE-READONLY OPTIONAL
  VALUE(I_REMOTE_CALL) TYPE SBIWA_FLAG DEFAULT SBIWA_C_FLAG_OFF
  TABLES
    I_T_SELECT TYPE SRSC_S_IF_SIMPLE-T_SELECT OPTIONAL
    I_T_FIELDS TYPE SRSC_S_IF_SIMPLE-T_FIELDS OPTIONAL
    E_T_DATA STRUCTURE ZNVBAK1 OPTIONAL
* EXCEPTIONS
  NO_MORE_DATA
  ERROR_PASSED_TO_MESS_HANDLER

**Example: DataSource for table SFLIGHT**

TABLES: ZNVBAK1.

* Auxiliary Selection criteria structure
DATA: L_S_SELECT TYPE SRSC_S_SELECT.

* Maximum number of lines for DB table
STATICS: S_S_IF TYPE SRSC_S_IF_SIMPLE,

* counter
  S_COUNTER_DATAPAKID LIKE SY-TABIX,

* cursor
  S_CURSOR TYPE CURSOR.

* Select ranges
  RANGES: L_R_CARRID FOR SFLIGHT-CARRID,
  L_R_CONNID FOR SFLIGHT-CONNID.
  RANGES: SDNO FOR ZNVBAK1-VBELN.

* Initialization mode (first call by SAPI) or data transfer mode
* (following calls) ?
  IF I_INITFLAG = SBIWA_C_FLAG_ON.

* Initialization: check input parameters
* buffer input parameters
* prepare data selection

* Check DataSource validity
CASE I_DSOURCE.
  WHEN 'ZFMEDS4'.
  WHEN OTHERS.
    IF 1 = 2. MESSAGE E009(R3). ENDIF.
  END_CASE.

* this is a typical log call. Please write every error message like this
LOG_WRITE 'E'                  
' message type
'R3'                  
' message class
'009'                  
' message number
I_DSOURCE   ' message variable 1
' '.                 
' message variable 2
RAISE ERROR_PASSED_TO_MESS_HANDLER.
ENDCASE.

APPEND LINES OF I_T_SELECT TO S_S_IF-T_SELECT.

* Fill parameter buffer for data extraction calls
S_S_IF-REQUNR = I_REQUNR.
S_S_IF-DSOURCE = I_DSOURCE.
S_S_IF-MAXSIZE = I_MAXSIZE.

* Fill field list table for an optimized select statement
* (in case that there is no 1:1 relation between InfoSource fields
* and database table fields this may be far from being trivial)
APPEND LINES OF I_T_FIELDS TO S_S_IF-T_FIELDS.

ELSE.
  *Initialization mode or data extraction ?
************************************************************************

* Data transfer: First Call
OPEN CURSOR + FETCH
Following Calls FETCH only
************************************************************************

* First data package -> OPEN CURSOR
IF S_COUNTER_DATAPAKID = 0.

* Fill range tables BW will only pass down simple selection criteria
  of the type SIGN = 'I' and OPTION = 'EQ' or OPTION = 'BT'.
  LOOP AT S_S_IF-T_SELECT INTO L_S_SELECT WHERE FIELDNM = 'CARRID'.
  MOVE-CORRESPONDING L_S_SELECT TO L_R_CARRID.
  APPEND L_R_CARRID.
  ENDLOOP.

  LOOP AT S_S_IF-T_SELECT INTO L_S_SELECT WHERE FIELDNM = 'CONNID'.
  MOVE-CORRESPONDING L_S_SELECT TO L_R_CONNID.
  APPEND L_R_CONNID.
  ENDLOOP.

  LOOP AT S_S_IF-T_SELECT INTO L_S_SELECT WHERE FIELDNM = 'VBELN'.
  MOVE-CORRESPONDING L_S_SELECT TO SDNO.
  APPEND SDNO.
  ENDLOOP.

  * Determine number of database records to be read per FETCH statement
  * from input parameter I_MAXSIZE. If there is a one to one relation
  * between DataSource table lines and database entries, this is trivial.
  * In other cases, it may be impossible and some estimated value has to
  * be determined.
  OPEN CURSOR WITH HOLD S_CURSOR FOR
  * SELECT (S_S_IF-TIELDS) FROM SFLIGHT
  * WHERE CARRID IN L_R_CARRID AND
  *       CONNID IN L_R_CONNID.
SELECT VBAK~VBELN VBAK~ERDAT VBAK~NETWR VBAP~MATNR
VBAP~POSNR
FROM VBAK
INNER JOIN VBAK on VBAK~VBELN = VBAP~VBELN
WHERE VBAK~VBELN IN SDNO.
  ENDIF.

  *First data package ?

* Fetch records into interface table.
  * named E_T_'Name of extract structure'.
FETCH NEXT CURSOR S_CURSOR
    APPENDING CORRESPONDING FIELDS
    OF TABLE E_T_DATA
    PACKAGE SIZE S_S_IF-MAXSIZE.

    IF SY-SUBRC <> 0.
        CLOSE CURSOR S_CURSOR.
        RAISE NO_MORE_DATA.
    ENDF.

    S_COUNTER_DATAPAKID = S_COUNTER_DATAPAKID + 1.
    ENDF.  "Initialization mode or data extraction ?
ENDFUNCTION.

Now Save, Activate the function module.

Goto SE80 -> Select Function Group, Activate the Function Group.

Note:

If you do not activate function Group, You will not be able to create data source in RS02

Data Source Creation

Then you have to create the data source, the data source name must be the name given in the source code of the Function module.

RS02 -> Create
Select Extr By Fun Module, Give name of Appl comp, Function Module, Extract Structure, save

Checking Extraction

Now give the selection Conditions and save.

RSA3 ->Give data source name and check extraction
BW Side.

1. RSA1 -> Source system, Src sys -> Rt Click, Data source Overview, Appl Comp, Rt click, Replicate.
2. RSA1 -> Info source -> Create Appl comp -> Rt clk, Assign Data source -> Src Sys

Some Blank Value will come in Info object; just press F4 and select, same field option
Then Press <- and save, Activate Data source.

Then From Info area, Create Info cube , Assign the Info source name created to char and Key fig of Cube

Cube will have no char in Char tab or on any other tab, Press Info source icon in Cube
If you double click that info source, every char. Key fig from that info source will come automatically into cube, just assign Dimension to Cube and save and activate.

Create Update Rule for the Cube based on the info source
Now Schedule the Extraction

The final data will be like this in Monitor
Monitor - Administrator Workbench

- Project: REGU_CONSUMERDATA DESTROY NWFL1
- Start Time: 04.01.2007 05:31:18
- Runtime: 7m 31s = 100.00% of the previous m.
- Application: Func Mod BPR from D&F
- InfoSource: ZMEDI_2_1 (Main Key Fig)
- InfoPackage: IP_FED (ZPACK-SAPFEKZP011)
- Data Targets: FED-CUBE2
- Processing: PSA and data target in series
- Update Mode: Full update
- Posting Method: Always update transaction data
- Records Check: Terminates if error occurs. Appropriate.

Monitor Successful (1)
- ZMEDI_2_1 (Main Key Fig)
- 04.01.2007
- 05:31:18 (18102/From 18102 Records)
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.