ABAP Code - Read SAP BW Infoprovider (Infocube, DSO, Multiprovider) Data using ABAP Code

Applies to:

This article is applicable to all the SAP BI 7.0 consultants who are accustomed with SAP ABAP skills.

Summary

This document specifies the detailed understanding of using the standard SAP BI function modules to fetch the Infoprovider (DSO, Infocube, and Multiprovider) data.

Author: Suraj TiggaEDW

Company: Capgemini Consulting India Pvt. Ltd.

Created on: 8 July 2010

Author Bio

Suraj Tigga is a Senior SAP BI / ABAP consultant at Capgemini Consulting, India. Suraj joined Capgemini Consulting in 2008 and has worked on multiple SAP BI implementation and support projects.

Table of Contents

Scenario

To implement a ABAP code which gets the data from Multiprovider (Built on DSOs and Infocube) based on Function Module RSDRI_INFOPROV_READ.While fetching the data the data packets should be taken into consideration.

Step-by-Step Solution

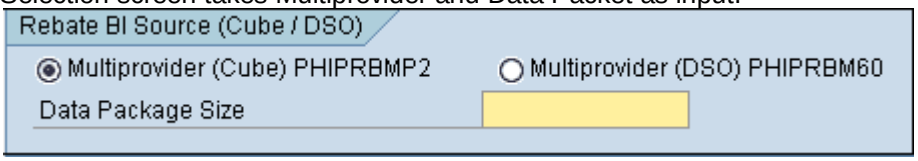
Solution is defined with two sections:

Selection Screen: Contains the Input Selection Screen

ABAP Code (Logic): The ABAP Code logic to fetch the data from Infoproviders

Selection Screen

Selection screen takes Multiprovider and Data Packet as input.



Multiprovider (Cube) PHIPRBMP2: Multipovider based on Info cubes

Multiprovider (DSO) PHIPRBM60: Multiprovider based on DSOs

Data Packet Size: The data packet size when the data is fed into DSO

ABAP Code (Logic)

Basic parameters of Function Module 'RSDRI_INFOPROV_READ':

Import Parameters:

I_INFOPROV: Technical name of Infoprovider (Infocubes, DSOs, Multiproviders)

I_TH_SFC: Characteristics of the Infoprovider which to be returned in the output

I_TH_SFK: Key Figures of the Infoprovider which to be returned in the output

I_T_RANGE: Contains the list of conditions based on which the data is to be fetched

I_REFERENCE_DATE: Specifies the key date for which time-dependent values are to be assessed.

I_ROLLUP_ONLY: Should only data that has been rolled up into the aggregates be considered, or should the most current data be read, which is possibly only located in the fact table of the InfoCube?

If a table of requests was specified (i_t_requid), if i_rollup_only = true, it is truncated at the number of requests already rolled up into the aggregates.

I_T_REQUID: With this parameter, a user can define a restriction using the request for an InfoCube.

I_PACKAGESIZE: Size of Returned Data Package (Default Value: 1000)

Export Parameters:

E_T_DATA: Internal Table with Query Result

E_END_OF_DATA: Internal Table with Query Result: Last Data Package Yes/No

Step1: Select the filer condition stored in a different DSO based on 'Customer', 'Chain ID' , 'National Group' and National Sub Group'. Store those values to internal table t_phsdrbse.

```
SELECT *
FROM /bic/aphsdrbse00
INTO TABLE t_phsdrbse.
```

Step2: Store the characteristics whose values are to be fetched from Multiprovider

```
********* CHARACTERISTIC *********************
  CLEAR: wa_s_sfc.
* Bill Num
  wa_s_sfc-chanm    = '0BILL_NUM'.
  wa_s_sfc-chaalias = 'BILL_NUM'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Bill Item
  wa_s_sfc-chanm    = '0BILL_ITEM'.
  wa_s_sfc-chaalias = 'BILL_ITEM'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Sold TO
  wa_s_sfc-chanm    = '0SOLD_TO'.
  wa_s_sfc-chaalias = 'SOLD_TO'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Material
  wa_s_sfc-chanm    = '0MATERIAL'.
  wa_s_sfc-chaalias = 'MATERIAL'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* PIP Type
  wa_s_sfc-chanm    = 'PIPIPTYPE'.
  wa_s_sfc-chaalias = '/BIC/PIPIPTYPE'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Postng Date
  wa_s_sfc-chanm    = '0PSTNG_DATE'.
  wa_s_sfc-chaalias = 'PSTNG_DATE'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Document Date
  wa_s_sfc-chanm    = 'PIPRDOCDT'.
  wa_s_sfc-chaalias = '/BIC/PIPRDOCDT'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Company Code
  wa_s_sfc-chanm    = '0COMP_CODE'.
  wa_s_sfc-chaalias = 'COMP_CODE'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* National Group
  wa_s_sfc-chanm    = 'PIPBRAN1'.
```

```
  wa_s_sfc-chaalias = '/BIC/PIPBRAN1'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* National Sub Group Code
  wa_s_sfc-chanm    = 'PIPBRAN2'.
  wa_s_sfc-chaalias = '/BIC/PIPBRAN2'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Region
  wa_s_sfc-chanm    = 'PIPBRAN3'.
  wa_s_sfc-chaalias = '/BIC/PIPBRAN3'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* District
  wa_s_sfc-chanm    = 'PIPBRAN4'.
  wa_s_sfc-chaalias = '/BIC/PIPBRAN4'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Submitted Customer Chain Identifier
  wa_s_sfc-chanm    = 'PIPCUSTCH'.
  wa_s_sfc-chaalias = '/BIC/PIPCUSTCH'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Customer Classification
  wa_s_sfc-chanm    = '0CUST_CLASS'.
  wa_s_sfc-chaalias = 'CUST_CLASS'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Dropship indicator
  wa_s_sfc-chanm    = 'ZSHDRPSHP'.
  wa_s_sfc-chaalias = '/BIC/ZSHDRPSHP'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Material group
  wa_s_sfc-chanm    = '0MATL_GROUP'.
  wa_s_sfc-chaalias = 'MATL_GROUP'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Contract Lead ID
  wa_s_sfc-chanm    = 'PSDPRTPID'.
  wa_s_sfc-chaalias = '/BIC/PSDPRTPID'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Contract Lead Indicator
  wa_s_sfc-chanm    = 'PIPCONLFL'.
  wa_s_sfc-chaalias = '/BIC/PIPCONLFL'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Netbill Indicator
  wa_s_sfc-chanm    = 'PIPNTBILL'.
  wa_s_sfc-chaalias = '/BIC/PIPNTBILL'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Original Invoice Number  on Returns
  wa_s_sfc-chanm    = 'PIPSFAKN'.
  wa_s_sfc-chaalias = '/BIC/PSDORGINV'.
```

```
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Sales document type
  wa_s_sfc-chanm    = '0DOC_TYPE'.
  wa_s_sfc-chaalias = 'DOC_TYPE'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Sales document
  wa_s_sfc-chanm    = '0DOC_NUMBER'.
  wa_s_sfc-chaalias = 'DOC_NUMBER'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* One-Stop Ind
  wa_s_sfc-chanm    = 'PMDONESTP'.
  wa_s_sfc-chaalias = '/BIC/PMDONESTP'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* RxPak Indicator
  wa_s_sfc-chanm    = 'PIPRXIND'.
  wa_s_sfc-chaalias = '/BIC/PIPRXIND'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Generics Rx Ind
  wa_s_sfc-chanm    = 'PIPGRXIND'.
  wa_s_sfc-chaalias = '/BIC/PIPGRXIND'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
* Matl Private Label Indicator
  wa_s_sfc-chanm    = 'PIPPRVLBL'.
  wa_s_sfc-chaalias = '/BIC/PIPPRVLBL'.
  wa_s_sfc-orderby  = 0.
  INSERT wa_s_sfc INTO TABLE t_th_sfc.
```

Step3: Store the Key Figures whose values are to be fetched from Multiprovider

```
  CLEAR wa_th_sfk.

* Miscellaneous Fees
  CLEAR wa_th_sfk.
  wa_th_sfk-kyfnm    = 'PIPMISFEE'.
  wa_th_sfk-kyfalias = '/BIC/PIPMISFEE'.
  wa_th_sfk-aggr     = 'SUM'.
  INSERT wa_th_sfk INTO TABLE t_th_sfk.

  CLEAR wa_th_sfk.
  wa_th_sfk-kyfnm    = 'PIPNETWR'.
  wa_th_sfk-kyfalias = 'SUBTOTAL_1'.
  wa_th_sfk-aggr     = 'SUM'.
  INSERT wa_th_sfk INTO TABLE t_th_sfk.
```

Step4: Assemble the selection condition which was fetch in step1 into a internal table

```
******** SELECTION CRITERIA ********************
  LOOP AT t_phsdrbse INTO wa_phsdrbse.

* Posting Date
    CLEAR wa_th_range .
    wa_th_range-chanm  = '0PSTNG_DATE'.
```

```
    wa_th_range-sign = rs_c_range_sign-including.
    wa_th_range-compop  = rs_c_range_opt-between.
    wa_th_range-low  = wa_phsdrbse-/bic/phrbfr.
    wa_th_range-high = wa_phsdrbse-/bic/phrbto.
    APPEND wa_th_range TO t_th_range.
    CLEAR wa_th_range.
* Customer
    IF wa_phsdrbse-customer IS NOT INITIAL.
      wa_th_range-chanm  = '0SOLD_TO'.
      wa_th_range-sign = rs_c_range_sign-including.
      wa_th_range-compop  = rs_c_range_opt-equal.
      wa_th_range-low  = wa_phsdrbse-customer.
      APPEND wa_th_range TO t_th_range.
      CLEAR wa_th_range.
    ENDIF.
* Chain ID
    IF wa_phsdrbse-/bic/zcustchid IS NOT INITIAL.
      wa_th_range-chanm  = 'PIPCUSTCH'.
      wa_th_range-sign = rs_c_range_sign-including.
      wa_th_range-compop  = rs_c_range_opt-equal.
      wa_th_range-low  = wa_phsdrbse-/bic/zcustchid.
      APPEND wa_th_range TO t_th_range.
      CLEAR wa_th_range.
    ENDIF.
* National Grp
    IF wa_phsdrbse-/bic/zcnltgpc IS NOT INITIAL.
      wa_th_range-chanm  = 'PIPBRAN1'.
      wa_th_range-sign = rs_c_range_sign-including.
      wa_th_range-compop  = rs_c_range_opt-equal.
* COnversion Exit
      CALL FUNCTION 'CONVERSION_EXIT_ALPHA_INPUT'
        EXPORTING
          input  = wa_phsdrbse-/bic/zcnltgpc
        IMPORTING
          output = wl_nat_grp.
      IF sy-subrc EQ 0.
        wa_th_range-low  = wl_nat_grp.
      ENDIF.
      APPEND wa_th_range TO t_th_range.
      CLEAR: wa_th_range , wl_nat_grp.
    ENDIF.
* National Sub Grp
    IF wa_phsdrbse-/bic/zcntlsgpc IS NOT INITIAL.
      wa_th_range-chanm  = 'PIPBRAN2'.
      wa_th_range-sign = rs_c_range_sign-including.
      wa_th_range-compop  = rs_c_range_opt-equal.
* COnversion Exit
      CALL FUNCTION 'CONVERSION_EXIT_ALPHA_INPUT'
        EXPORTING
          input  = wa_phsdrbse-/bic/zcntlsgpc
        IMPORTING
          output = wl_nat_sub_grp.
      IF sy-subrc EQ 0.
        wa_th_range-low  = wl_nat_sub_grp.
      ENDIF.
      APPEND wa_th_range TO t_th_range.
```

```
      CLEAR: wa_th_range , wl_nat_sub_grp .
    ENDIF.
  ENDLOOP.
```

Sep5: Call the function module RSDRI_INFOPROV_READ using the values stored in different internal tables

```
If Multiprovider (DSO) is checked  then
   wl_infoprov = 'PHIPRBM60'.
```

If Multiprovider (Infocubes) is checked then,

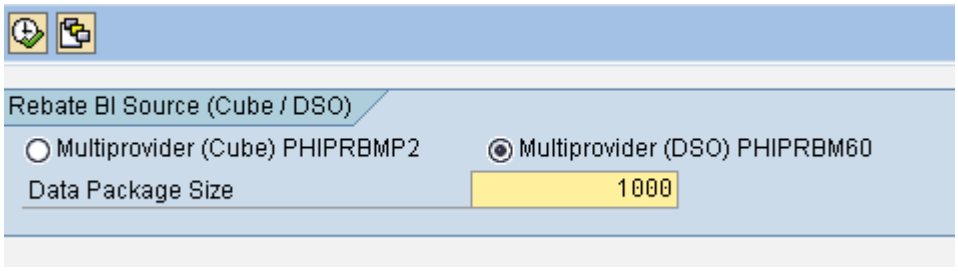
```
    wl_infoprov = 'PHIPRBMP2'.

  WHILE t_end_of_data = rs_c_false.
***** Function Mdoule to fetch data *****************
    CALL FUNCTION 'RSDRI_INFOPROV_READ'
      EXPORTING
        i_infoprov             = wl_infoprov
        i_th_sfc               = t_th_sfc
        i_th_sfk               = t_th_sfk
        i_t_range              = t_th_range
        i_reference_date       = sy-datum
        i_save_in_table        = rs_c_false
        i_save_in_file         = rs_c_false
        i_packagesize          = wl_data_pac
        i_authority_check      = rsdrc_c_authchk-read
      IMPORTING
        e_t_data               = t_e_t_data
        e_end_of_data          = t_end_of_data
      CHANGING
        c_first_call           = t_first_call
      EXCEPTIONS
        illegal_input          = 1
        illegal_input_sfc      = 2
        illegal_input_sfk      = 3
        illegal_input_range    = 4
        illegal_input_tablesel = 5
        no_authorization       = 6
        illegal_download       = 8
        illegal_tablename      = 9
        OTHERS                 = 11.
    IF sy-subrc EQ 0.
    ENDIF.
  ENDWHILE.
```

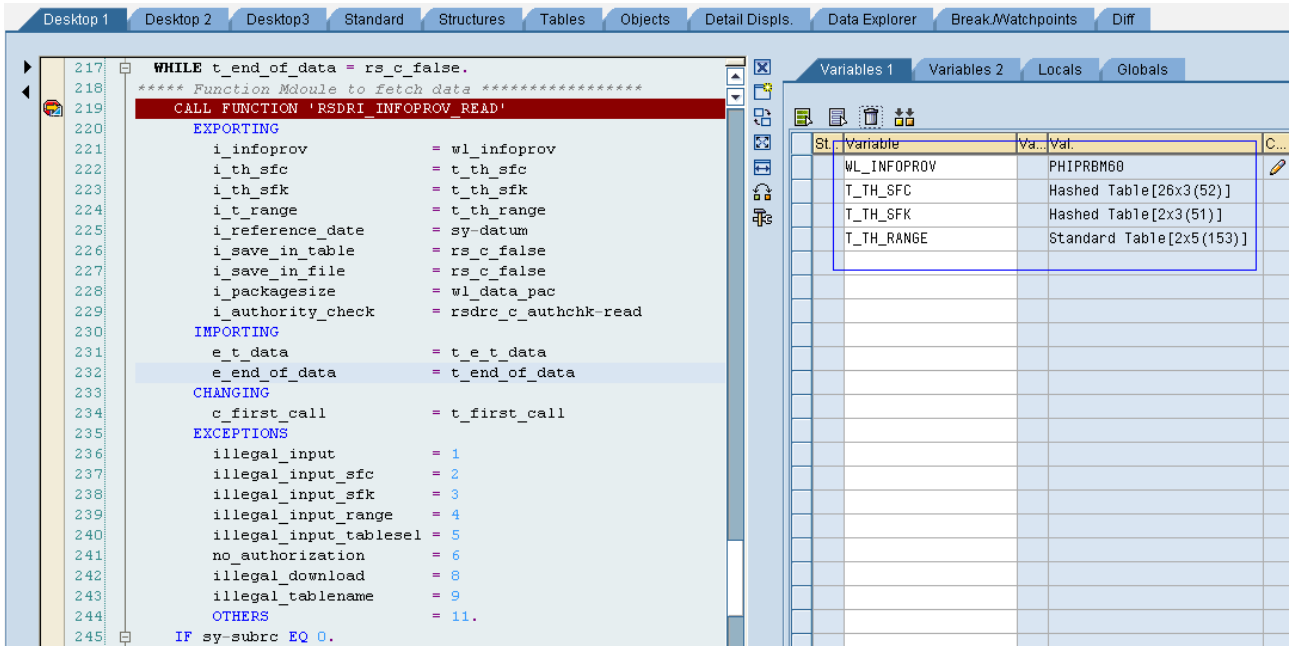
Execution

a) Selection Screen Input

Enter the values as shown in selection screen and execute

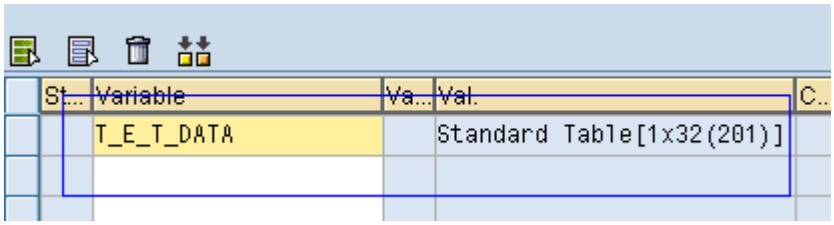


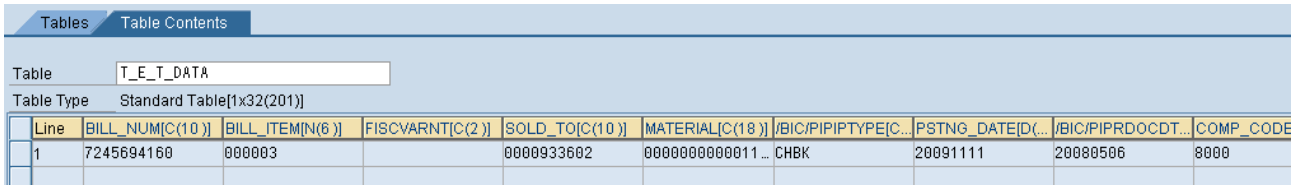
b) Values fetched from Function Module



Check the values in the internal tables for characteristics and Key Figures which are fed to the Function Module.

Query values are stored in internal table:





Notes:

c) When t_end_of_data is checked , then it reflects the last data packet is being processed

d) Internal table `t_e_t_data` contains all the data being fetched from the infoproviders

After the data is successfully fetched from the Function Module, update those values to the direct DSO.

Other Functions (Which can also be used to fetch Infoprovider related data):

a) RSDRI_INFOPROV_READ_RFC(BW Data Manager: General RFC Read Interface)

b) RSDRC_INFOCUBE_READ (Read Infocube Data)

c) RSSEM_INFOPROV_READ (Read transaction data to node selection)

d) RSDPL_CUBE_DATA_READ (DM-API: Read Transaction Data of a Cube)

e) RSDRI_AGGREGATE_DATA (Read Aggregated Data)

f) BAPI_CUBE_GETDETAIL (Read Infocube Details)

Important Function Groups: RSSEM2, RSDPL, RSDRC_INFOCUBE

Related Content

SDN Forum Link1

SDN Forum Link2

SDN Forum Link3

For more information, visit the EDW 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.