How to Automate Dimension Member Selection when Exporting BPC Transaction Data

Applies to:

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

Summary

SAP had provided a how to document that implements a custom process chain to export BPC transaction data to the connected application servers. But one has to manually choose the dimension members to identify the data that needs to be exported. This document describes the steps to automate dimension member selections when some business logic exists to do so. This method can be extended to automate the loads if it is desired to split the BPC transaction data into smaller subsets when the transactional data volume is very high

Author: Ramam Velivala

Company: Deloitte Consulting

Created on: 8th June 2011

Author Bio

Ramam Velivala is a Lead consultant with SAP BW/BI, BPC and EPM practice at Deloitte. He is currently leading the SAP-BW track integration with EPM tools. He has extensively worked on PCM modeling to extract data to SAP-BW and BPC systems. He has more than 12 years of SAP experience in providing analytic solution in various SAP functional modules like Finance, Controlling, Procurement etc.

Table of Contents

Business Scenario

In a typical business scenario, plan data in the BPC system is required to be exported to other systems, for example to other external reporting systems or ECC. SAP already provided a how to document on exporting BPC transaction data using a custom process chain. But with the existing SAP solution, one has to manually choose the dimension members to identify the data that needs to be exported. More often than not, the business wants this export to happen based on a pre-determined business logic so that the load process can be automated using process chains. A simple business logic is implemented in this document to show how to achieve this requirement. Also, if the data volume from BPC is huge, it may be desirable to split the data into smaller subsets and then export them separately.

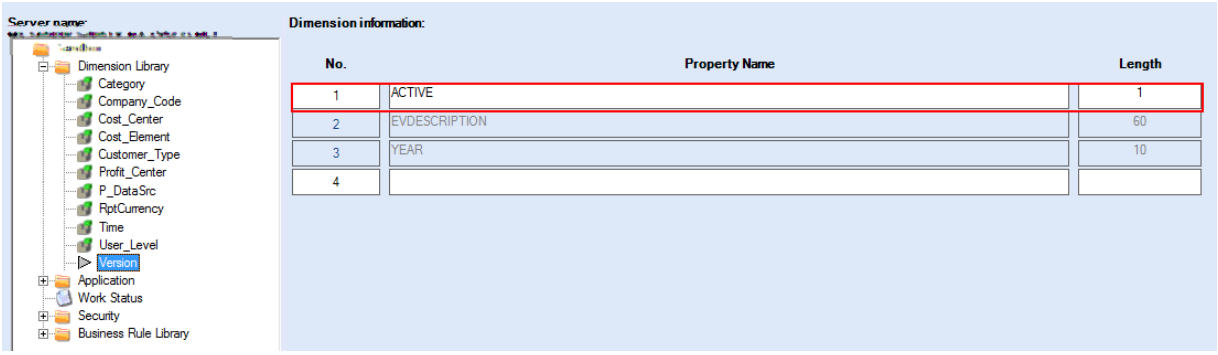
Existing SAP solution to export the BPC transaction data using a custom process chain can be accessed at the following link: http://www.sdn.sap.com/irj/bpx/go/portal/prtroot/docs/library/uuid/b0427318-5ffb-2b10-9cac-96ec44c73c11?QuickLink=index&overridelayout=true

Note: The following procedure needs to be implemented only after all the steps in the above mentioned SAP how-to-guide are performed successfully.

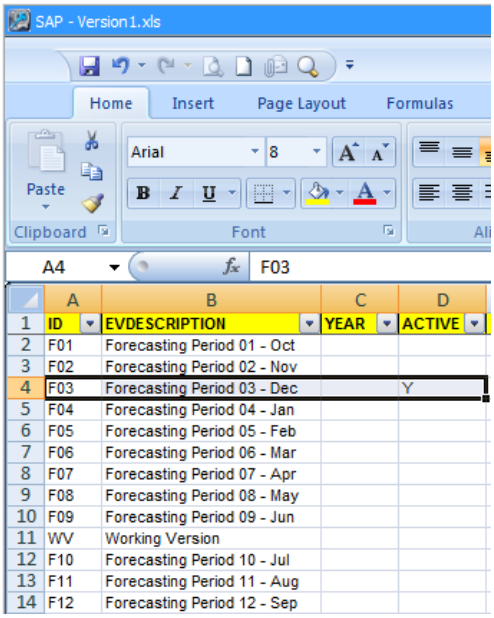
Step by Step Procedure

Say the business wants to export the BPC transaction data pertaining to the active version only.

To represent the active version, a property called “Active” is maintained on the “Version” dimension as shown below. The value “Y” against a dimension member represents the active version.

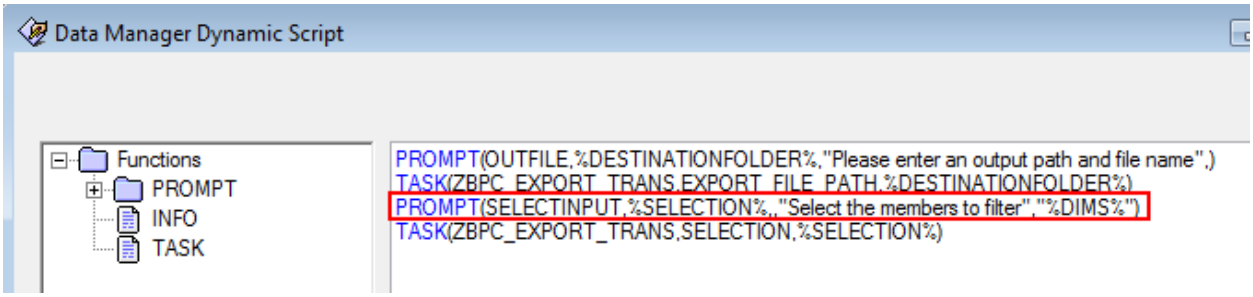


Following screenshot shows that F03 is the active version.

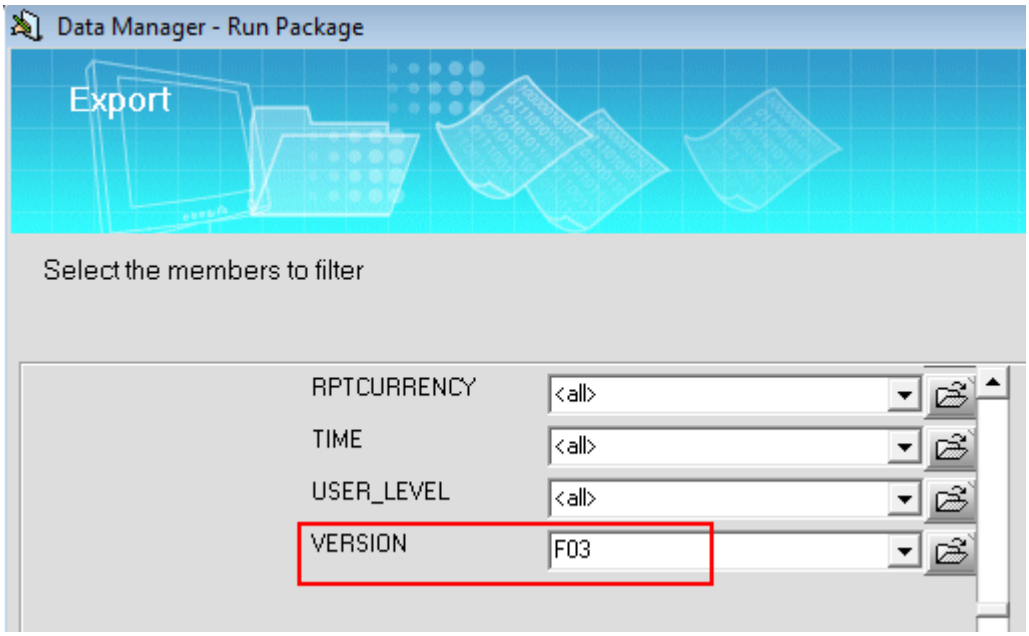


Existing Procedure

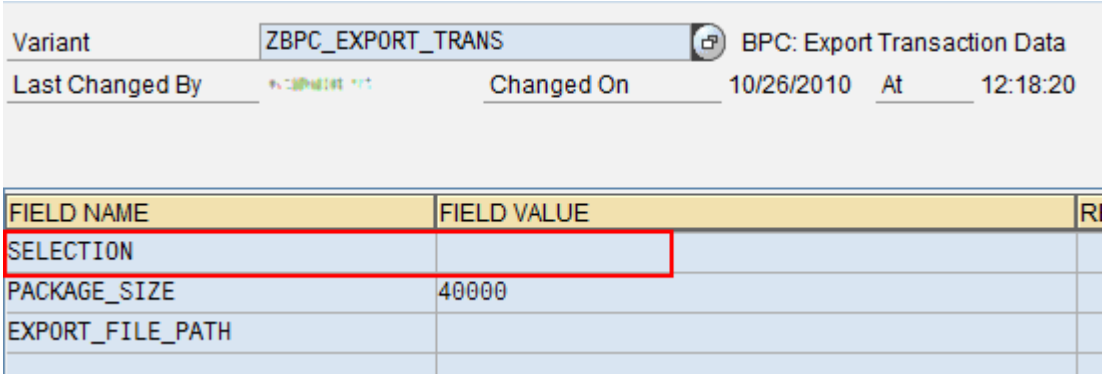
In the SAP delivered solution, the data manager package contains a call to the dimension member selection screen as shown below. To reach the following screen, follow eData -> Organize Packages. Right click "Export" package and choose "Modify Package". Click "View Package" and then click "Advanced".



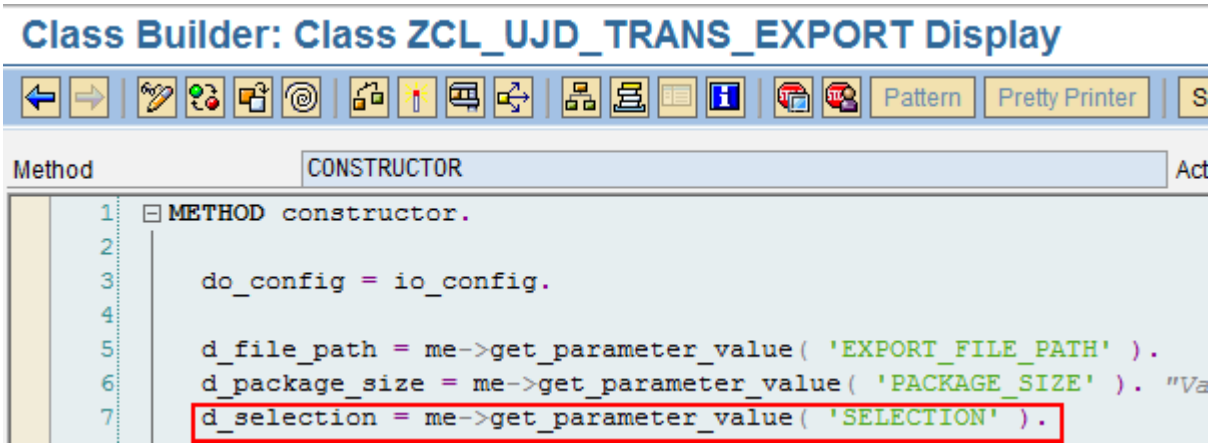
When the data manager package is run, the user has to manually choose the active version (F03 here) in the dimension member selection screen to achieve the business requirement.



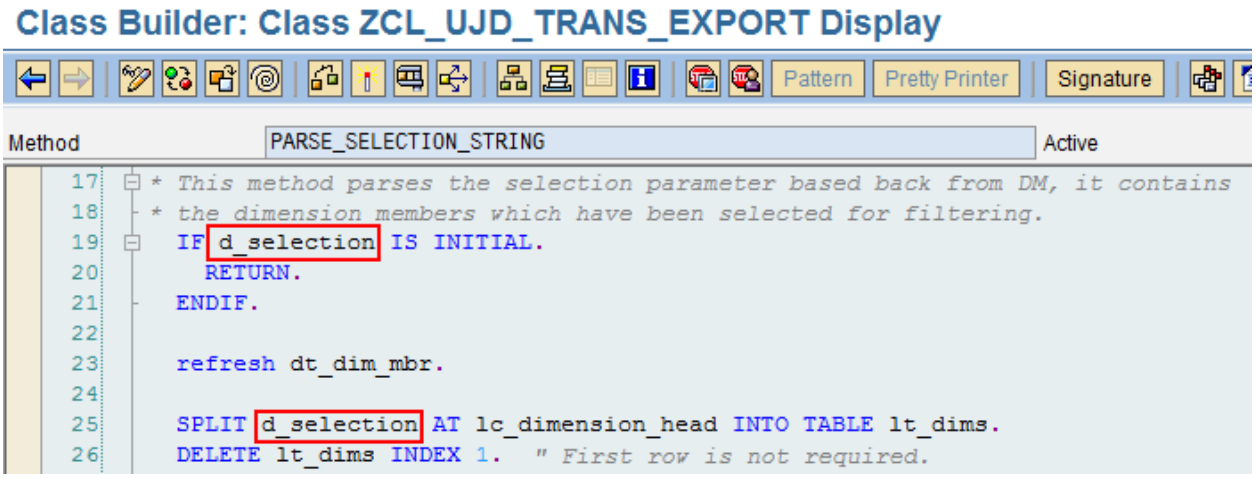
This selection is passed on to the "SELECTION" parameter in the underlying process chain.



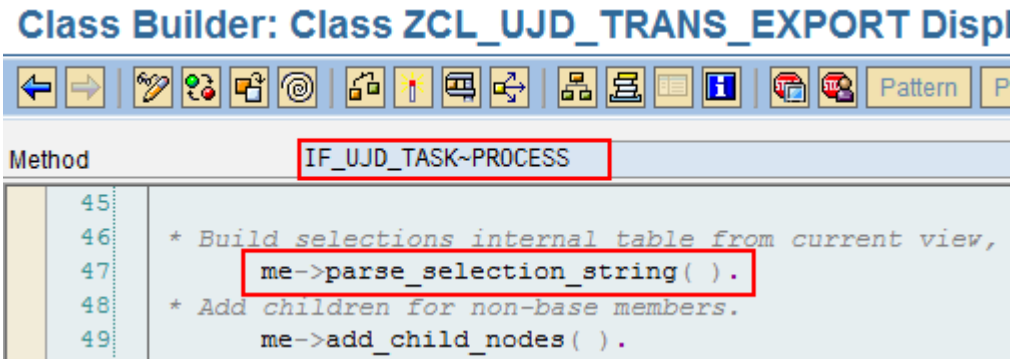
ZCL_UJD_TRANS_EXPORT is one of the classes delivered as part of the SAP how to guide. In the constructor method for this class, “SELECTION” parameter from the process chain is read into the attribute “D_SELECTION” of the class.



This attribute is used in the method PARSE_SELECTION_STRING to determine the individual dimension members that are to be used for filtering the transaction data to export.



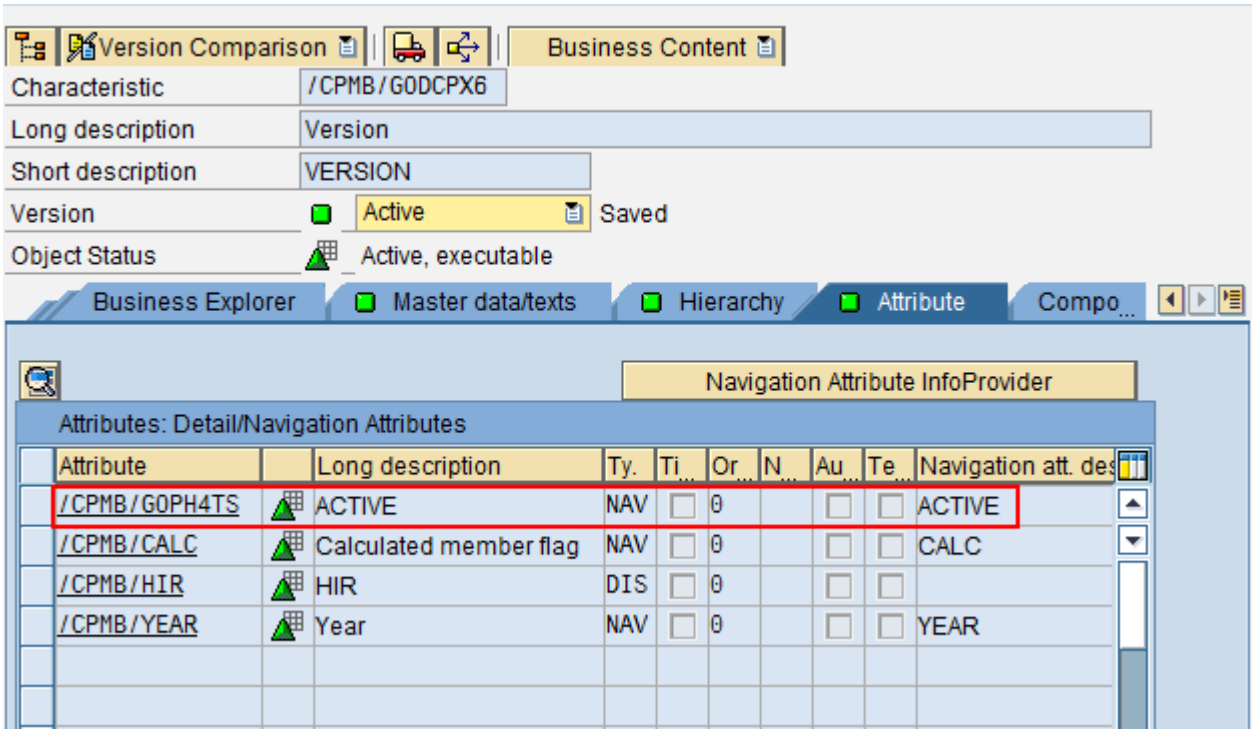
The method PARSE_SELECTION_STRING is called from the implementation of the interface method IF_UJD_TASK~PROCESS.



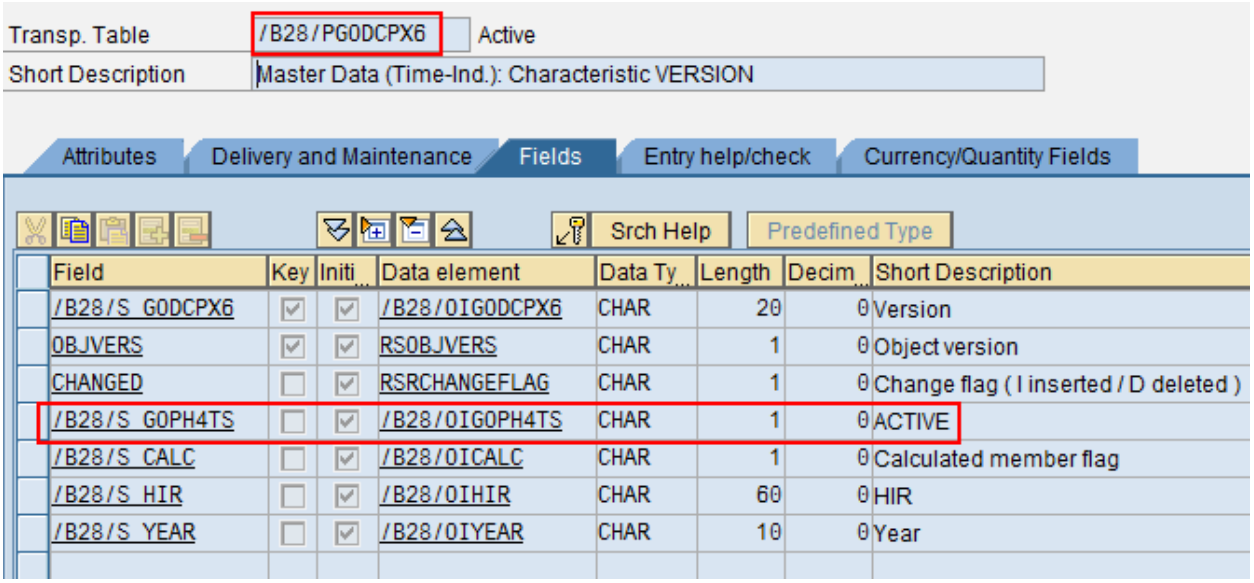
Since we want to do away with the manual selection, the suggested solution below will not pass any value to the attribute “D_SELECTION”. Instead, it implements a custom method that will be called right after the call to PARSE_SELECTION_STRING in the above.

Custom Dimension Member Selection

Following screenshot shows the details of "Version" InfoObject in BW.

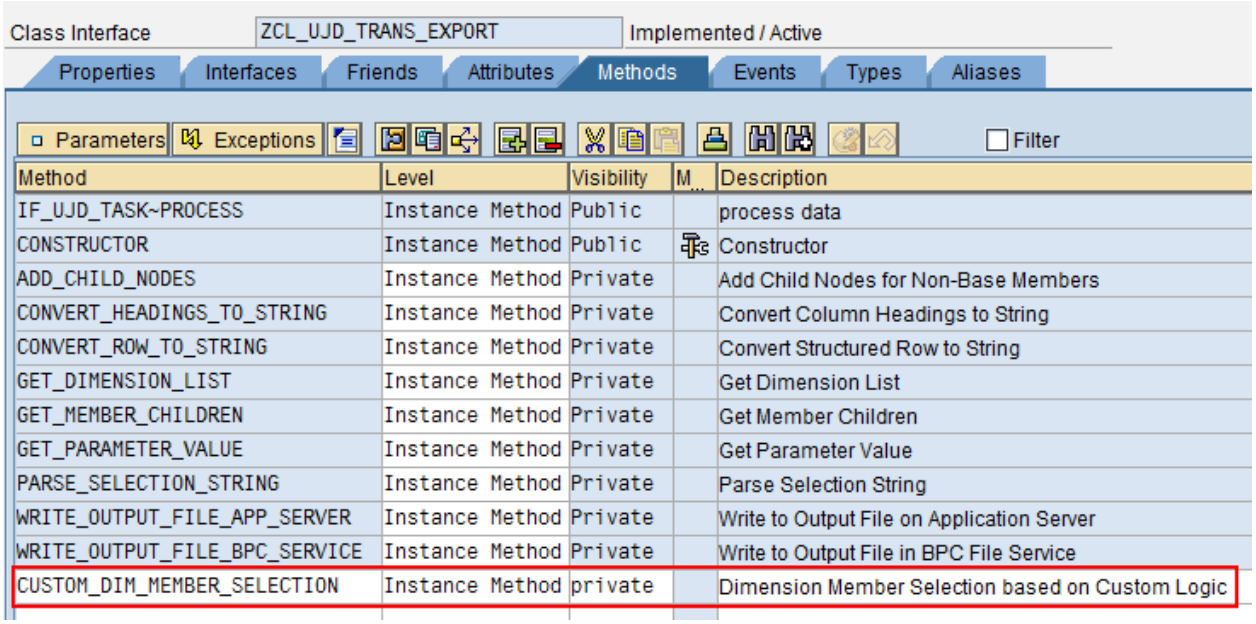


Following screenshot represents the SE11 view of Master data table for the above InfoObject.



We can implement the custom business logic in a new method and call it from IF_UJD_TASK~PROCESS.

Go to SE24 and open the class ZCL_UJD_TRANS_EXPORT. Go to "Methods" tab and click  . Enter the name of the new method, CUSTOM_DIM_MEMBER_SELECTION and make it "Private". Now click "Save".



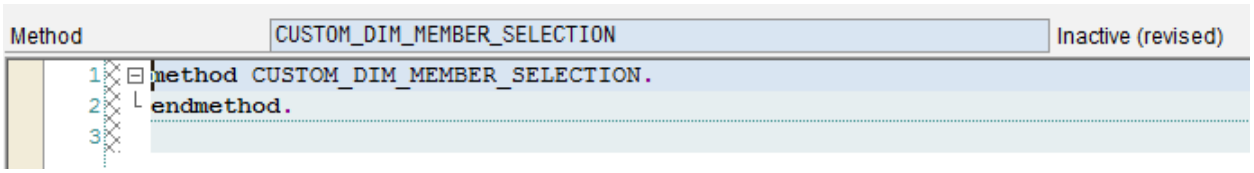
Class Interface: ZCL_UJD_TRANS_EXPORT — Implemented / Active

Properties | Interfaces | Friends | Attributes | Methods | Events | Types | Aliases

Parameters | Exceptions | Filter

Method	Level	Visibility	M...	Description
IF_UJD_TASK~PROCESS	Instance Method	Public		process data
CONSTRUCTOR	Instance Method	Public		Constructor
ADD_CHILD_NODES	Instance Method	Private		Add Child Nodes for Non-Base Members
CONVERT_HEADINGS_TO_STRING	Instance Method	Private		Convert Column Headings to String
CONVERT_ROW_TO_STRING	Instance Method	Private		Convert Structured Row to String
GET_DIMENSION_LIST	Instance Method	Private		Get Dimension List
GET_MEMBER_CHILDREN	Instance Method	Private		Get Member Children
GET_PARAMETER_VALUE	Instance Method	Private		Get Parameter Value
PARSE_SELECTION_STRING	Instance Method	Private		Parse Selection String
WRITE_OUTPUT_FILE_APP_SERVER	Instance Method	Private		Write to Output File on Application Server
WRITE_OUTPUT_FILE_BPC_SERVICE	Instance Method	Private		Write to Output File in BPC File Service
CUSTOM_DIM_MEMBER_SELECTION	Instance Method	private		Dimension Member Selection based on Custom Logic

Now double click on CUSTOM_DIM_MEMBER_SELECTION to reach the following screen.



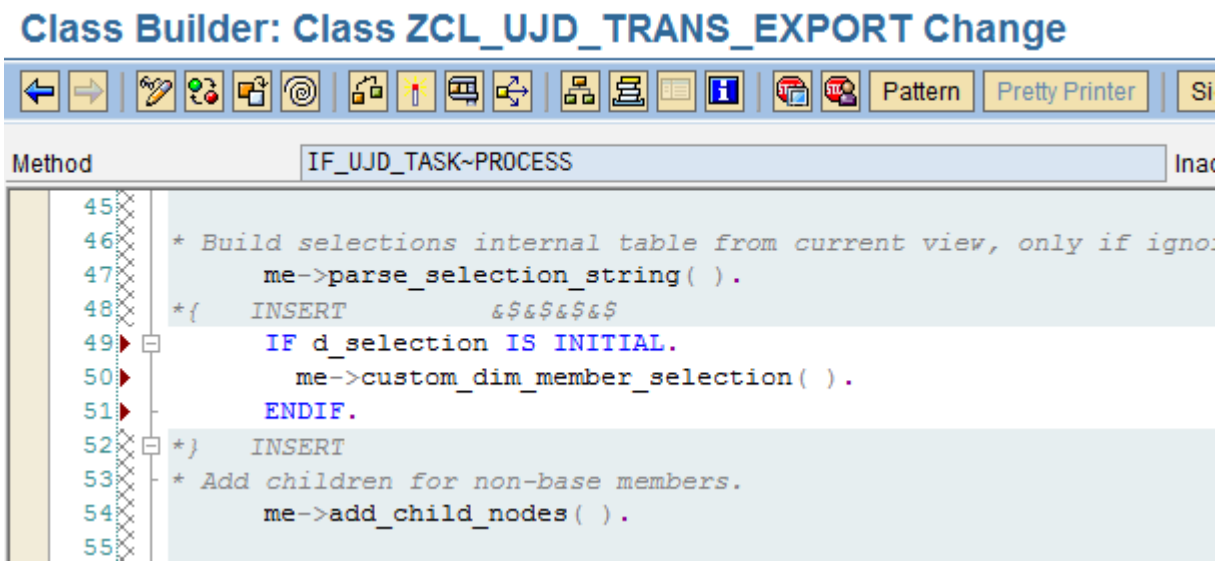
Method: CUSTOM_DIM_MEMBER_SELECTION — Inactive (revised)

```
1  method CUSTOM_DIM_MEMBER_SELECTION.
2  endmethod.
3
```

Place the cursor on the second line and click Insert. Here you can write the logic that suits your business requirement. Following is the code to obtain the active version. Save the code.

```
1  method CUSTOM_DIM_MEMBER_SELECTION.
2  *{   INSERT         &$&$&$&$                                    1
3    DATA: lv_actver TYPE /B28/OIGODCPX6.
4
5    FIELD-SYMBOLS: <ls_dim_mbr> LIKE LINE OF dt_dim_mbr.
6
7    refresh dt_dim_mbr.
8
9  * Reading the VERSION Dimension with property ACTIVE = 'Y' to obtain the "Active Version"
10   SELECT SINGLE /B28/S_GODCPX6
11     FROM /B28/PGODCPX6
12     INTO lv_actver
13     WHERE /B28/S_GOPH4TS = 'Y'.
14
15   APPEND INITIAL LINE TO dt_dim_mbr ASSIGNING <ls_dim_mbr>.
16   <ls_dim_mbr>-dimension = 'VERSION'.
17   <ls_dim_mbr>-member    = lv_actver.
18 *}   INSERT
19 endmethod.
```

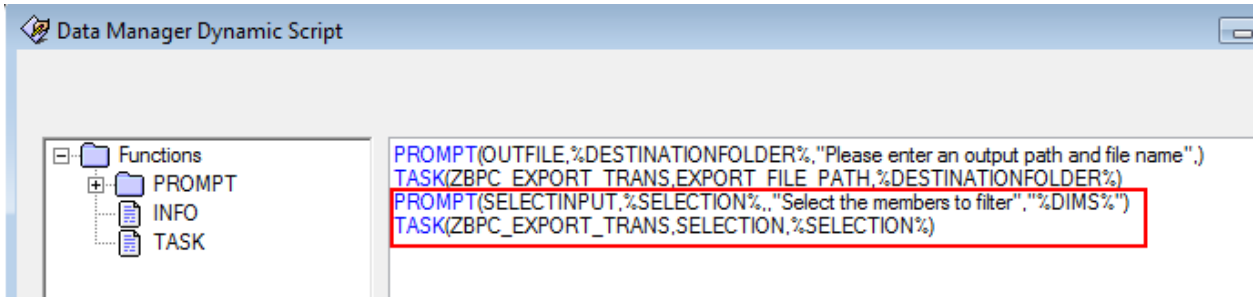
Now double click on IF_UJD_TASK~PROCESS. Call the method CUSTOM_DIM_MEMBER_SELECTION as shown below. Our custom method is called only when D_SELECTION is empty. Save the method and activate the class.



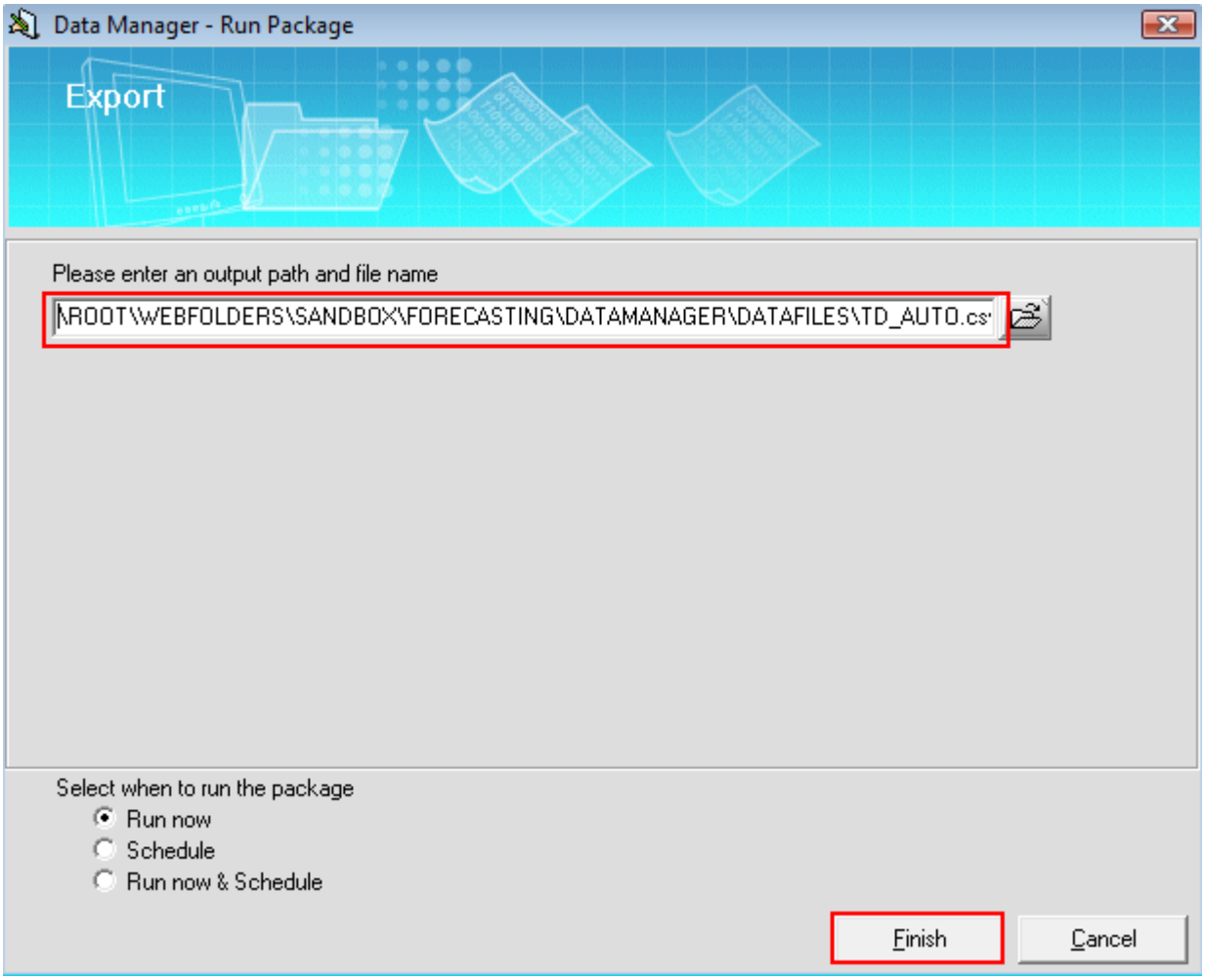
Note that the call to method PARSE_SELECTION_STRING in the above is useless as D_SELECTION is empty (see the steps below to remove the dimension selection screen option from DM package). Yet the call needs to be made to the method as the same class is used in other process types.

Now the call to dimension member selection screen needs to be removed. To do that, go to eData -> Organize Packages. Right click “Export” package and choose “Modify Package”. Click “View Package” and then click “Advanced”.

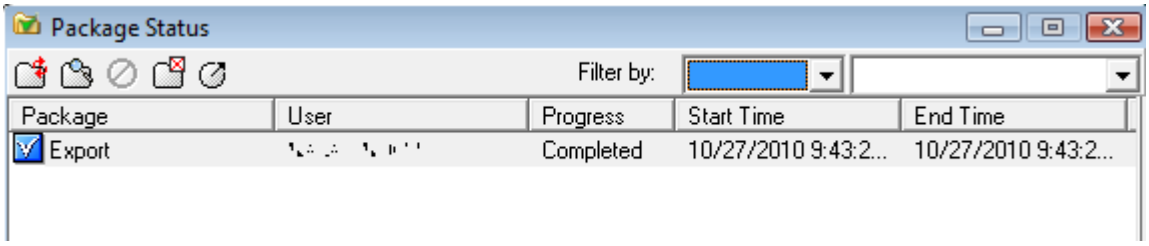
Remove the following piece of code from the data manager package.



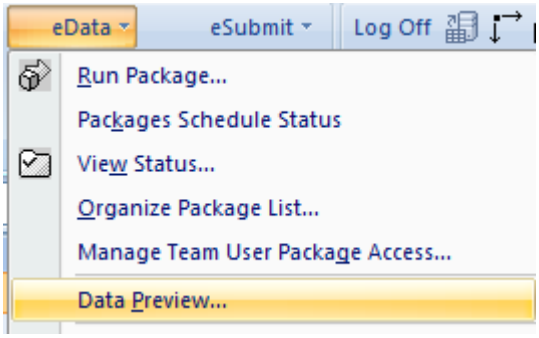
Now execute the data manager package to see that there is no selection for dimension members. All you can select is the file path that you want to write to.

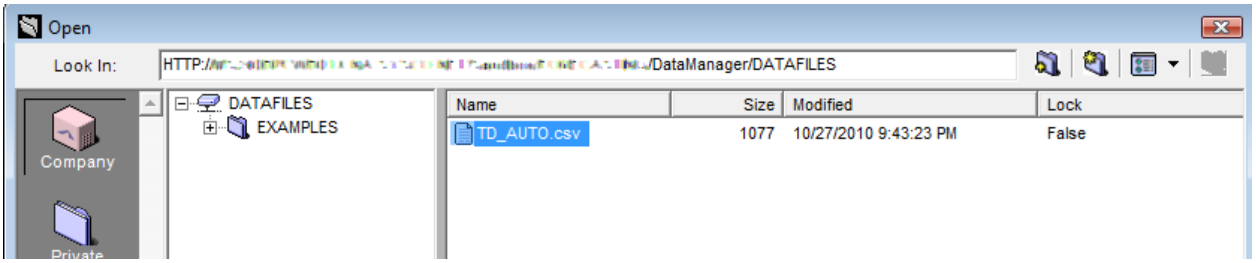


Check the package status.

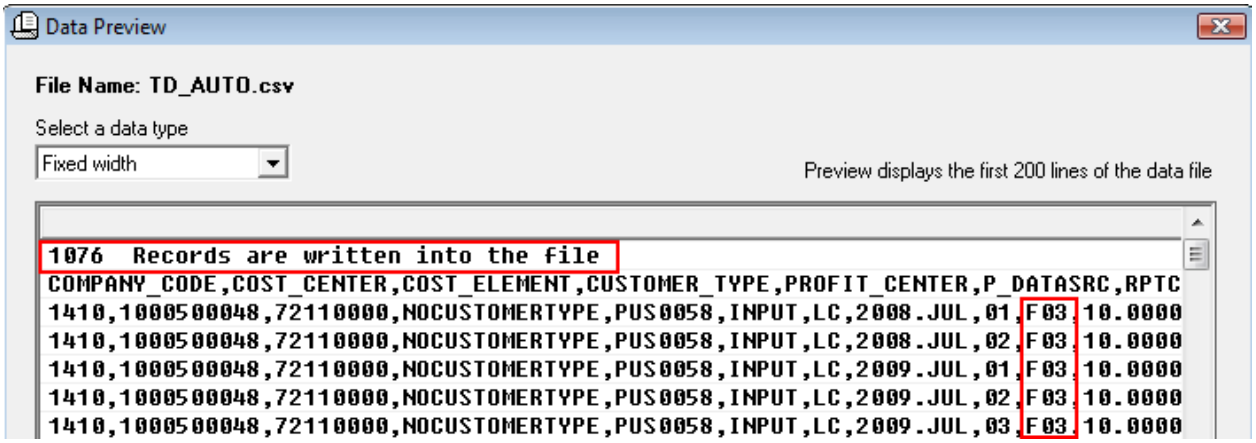


If it is successful then go to eData -> Data Preview to verify the data.

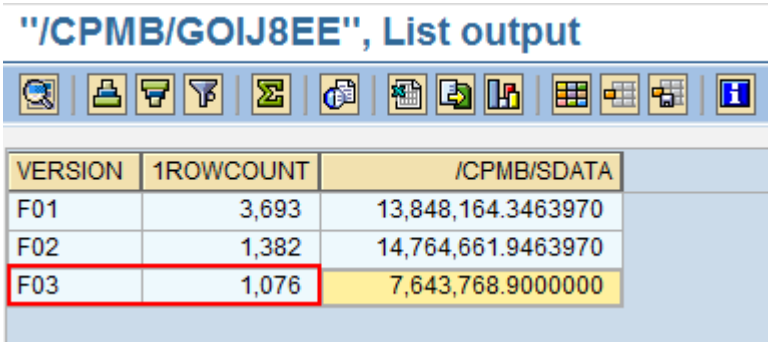




Double click the file to preview the data.



There are 1076 records written into the file, which matches with the number of records in the LISTCUBE screenshot below for the corresponding BPC application.



VERSION	1ROWCOUNT	/CPMB/SDATA
F01	3,693	13,848,164.3463970
F02	1,382	14,764,661.9463970
F03	1,076	7,643,768.9000000

Note: SAP delivered code does not write the record count to the file. So a minor modification is made to the code to show the record count for validation purposes.

Appendix

Code in the Data Manager Package

```
PROMPT(OUTFILE,%DESTINATIONFOLDER%,"Please enter an output path and file name",)
TASK(ZBPC_EXPORT_TRANS,EXPORT_FILE_PATH,%DESTINATIONFOLDER%)
```

Code in the method CUSTOM_DIM_MEMBER_SELECTION

```
method CUSTOM_DIM_MEMBER_SELECTION.
*{   INSERT          1
  DATA: lv_actver TYPE /B28/OIGODCPX6.

  FIELD-SYMBOLS: <ls_dim_mbr> LIKE LINE OF dt_dim_mbr.

  refresh dt_dim_mbr.

* Reading the VERSION Dimension with property ACTIVE = 'Y' to obtain the "Active Version"
  SELECT SINGLE /B28/S_GODCPX6
    FROM /B28/PGODCPX6
    INTO lv_actver
    WHERE /B28/S_GOPH4TS = 'Y'.

  APPEND INITIAL LINE TO dt_dim_mbr ASSIGNING <ls_dim_mbr>.
  <ls_dim_mbr>-dimension = 'VERSION'.
  <ls_dim_mbr>-member    = lv_actver.
*}   INSERT
endmethod.
```

Code in the method IF_UJD_TASK~PROCESS (from line 49).

```
IF d_selection IS INITIAL.
   me->custom_dim_member_selection( ).
ENDIF.
```

Related Content

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.