Sample Code for Expert Routine

Abstract:

This Content provides initial guidance & help in writing expert routine in transformations of BI7.0. Also provides brief information of how to use & where to use expert routine, along with few examples of scenarios and a sample code based on a scenario.

Description:

This content aims at:

1. Giving basic know how about expert routines in BI 7.0

Summary:

1. With expert routines, unlike start routine and end routine, the system will provide you with internal table of the type of source and target namely SOURCE_PACKAGE and RESULT_PACKAGE. Your main aim will be to fill up the RESULT_PACKAGE. In between you can apply your own logic. But you should fill up the RESULT_PACKAGE internal table. Both internal tables will be given as import and export parameter of a method.
2. To access fields from source fields, SOURCE_PACKAGE type _ty_t_SC_1
3. To access fields of result fields, RESULT_PACKAGE type _ty_t_TG_1
4. Location of the code: expert routine is written in transformations of data targets such as DSO, master data, OHTs etc.

5. Following is the screen shot of such transformation:

Introduction:

This type of routine is used in special cases only. You can use the expert routine if there are not sufficient functions to perform a transformation. The expert routine should be used as an acting resolution until the required functionality becomes available in the standard routine. You can use this to program the transformation yourself without using the available rule types. You must implement the message transfer to the monitor yourself. If you have already created transformation rules, the system deletes them once you have created an expert routine. If the target of the transformation is a Data Store object, key figures are updated by default with the aggregation behavior Overwrite (MOVE).

Use of Expert routines:

If you don't want to go in for the system generated program for the transformation and would prefer to write the entire code on your own i.e. start routine + transformation + end routine you should go for an expert routine.

The data flow now goes as
Raw data -> expert routine -> info provider.

Expert routine: This is the strongest and most complex of all routines. This allows you to design the entire data flow of the transformation. If you create an Expert routine the system will automatically delete all the field level mapping, Start routine and end routine. You also have to take care of the error handling and monitoring.
Expert Routine will have both Source and Target Fields. In Expert Routines u won't have mapping between source and target fields.
Transformation is done through coding in expert routine.

Scenario:

We are loading salary details for employees from a flat file into a DSO.
Following sample code is written for displaying salary details along with different calculations related to salary. Ex. Gross salary etc.

Sample Code:

PROGRAM trans_routine.
*---------------------------------------------------------------------*
*       CLASS routine DEFINITION*
*---------------------------------------------------------------------*
CLASS lcl_transform DEFINITION.
PUBLIC SECTION.
*  Attributs
DATA:
  p_check_master_data_exist
  TYPE Rsodsocheckonly READ-ONLY,
*  Instance for getting request runtime attributs;
*  Available information: Refer to methods of
*  interface 'if_rsbk_request_admintab_view'


PRIVATE SECTION.

TYPE-POOLS: rsd, rstr.

* Rule specific types

TYPES:

BEGIN OF _ty_s_SC_1,
  * Field: /BIC/YEM_NUM EMP NUMBER.
    /BIC/YEM_NUM TYPE C LENGTH 10,
  * Field: /BIC/YEM_FNM EMP FIRST NAME.
    /BIC/YEM_FNM TYPE C LENGTH 25,
  * Field: /BIC/YEM_LNM EMP LAST NAME.
    /BIC/YEM_LNM TYPE C LENGTH 25,
  * Field: /BIC/YEM_LOC EMP LOCATION.
    /BIC/YEM_LOC TYPE C LENGTH 25,
  * Field: /BIC/YEM_PH1 Emp Phone number1.
    /BIC/YEM_PH1 TYPE N LENGTH 15,
  * Field: /BIC/YEM_PH2 Emp Phone number2.
    /BIC/YEM_PH2 TYPE N LENGTH 15,
  * Field: /BIC/YEM_PH3 Emp Phone number3.
    /BIC/YEM_PH3 TYPE N LENGTH 15,
  * Field: /BIC/YEM_DSG EMP DESIGNATION.
    /BIC/YEM_DSG TYPE C LENGTH 10,
  * Field: /BIC/YEM_JB EMP JOB BAND.
    /BIC/YEM_JB TYPE C LENGTH 5,
  * Field: /BIC/YEM_CRR EMP CRR RATING.
    /BIC/YEM_CRR TYPE N LENGTH 5,
  * Field: /BIC/YEM_BSAL EMP BASIC SALARY.
    /BIC/YEM_BSAL TYPE P LENGTH 9 DECIMALS 2,
  * Field: CURRENCY Currency.
    CURRENCY TYPE C LENGTH 5,
  * Field: /BIC/YEM_DA EMP DEARNESS ALLOWANCE.
    /BIC/YEM_DA TYPE P LENGTH 9 DECIMALS 2,
  * Field: /BIC/YEM_BOA EMP BOA.
    /BIC/YEM_BOA TYPE P LENGTH 9 DECIMALS 2,
  * Field: /BIC/YEM_GRS EMP MONTHLY GROSS SALARY.
    /BIC/YEM_GRS TYPE P LENGTH 9 DECIMALS 2,
  * Field: /BIC/YEM_PF EMP PROVIDENT FUND.
    /BIC/YEM_PF TYPE P LENGTH 9 DECIMALS 2,
  * Field: /BIC/YEM_GRT EMP GRATUITY.
    /BIC/YEM_GRT TYPE P LENGTH 9 DECIMALS 2,
  * Field: /BIC/YEM_TGRS EMP TOTAL GROSS SALARY.
    /BIC/YEM_TGRS TYPE P LENGTH 9 DECIMALS 2,
  * Field: /BIC/YEM_PINC EMP PERFORMANCE INCOME.
    /BIC/YEM_PINC TYPE P LENGTH 9 DECIMALS 2,
  * Field: /BIC/YEM_GRINC EMP TOTAL GROSS INC.
    /BIC/YEM_GRINC TYPE P LENGTH 9 DECIMALS 2,
  * Field: /BIC/YEM_REMGR REPORTING MANAGER.
    /BIC/YEM_REMGR TYPE C LENGTH 10,
  * Field: RECORD Record Number.
    RECORD TYPE RSARECORD,
END OF _ty_s_SC_1.

TYPES:

BEGIN OF _ty_s_SC_1
  WITH NON-UNIQUE DEFAULT KEY.

TYPES:

BEGIN OF _ty_s_SC_1
* CLASS routine IMPLEMENTATION
*---------------------------------------------------------------------*
*---------------------------------------------------------------------*
CLASS lcl_transform IMPLEMENTATION.

*----------------------------------------------------------------------*
* Method expert_routine
*----------------------------------------------------------------------*
* Calculation of result package via expert routine
*----------------------------------------------------------------------*
* -> package of source segments
* <- result package
*----------------------------------------------------------------------*
METHOD expert_routine.

*** Segments ***

FIELD-SYMBOLS:
<SOURCE_FIELDS> TYPE _ty_s_SC_1.

DATA:
RESULT_FIELDS TYPE _ty_s_TG_1.

*$*$ begin of routine - insert your code only below this line
... "insert your code here
Data: gw_result_package TYPE _ty_s_SC_1.
Data: gw_result_package1 TYPE _ty_s_TG_1.

DATA: ZVAR1 TYPE I,
ZVAR2(10) TYPE n,
ZVAR3(10) TYPE n,
ZVAR4(10) TYPE n,
ZVAR5 TYPE c,
ZVAR6 TYPE c,
ZVAR7 TYPE I,
ZVAR8 TYPE I,
ZVAR9 TYPE I,
ZVAR10 TYPE I,
ZVAR11 TYPE I,
ZVAR12 TYPE I,
ZVAR13 TYPE I,
ZVAR14 TYPE I,
ZVAR15 TYPE I,
ZVAR16 TYPE I,
ZVAR17 TYPE I,

*Loop to read source package into internal Package and store values into local variables.

LOOP at SOURCE_PACKAGE into gw_result_package.

Insert code for expert routine here

ZVAR1 = gw_result_package-/BIC/YEM_NUM.
ZVAR2 = gw_result_package-/BIC/YEM_PH1.
ZVAR3 = gw_result_package-/BIC/YEM_PH2.
ZVAR4 = gw_result_package-/BIC/YEM_PH3.
ZVAR5 = gw_result_package-/BIC/YEM_DSG.
ZVAR6 = gw_result_package-/BIC/YEM_JB.
ZVAR7 = gw_result_package-/BIC/YEM_CRR.
ZVAR8 = gw_result_package-/BIC/YEM_BSAL.
ZVAR9 = gw_result_package-/BIC/YEM_DA.
ZVAR10 = gw_result_package-/BIC/YEM_BOA.
ZVAR11 = gw_result_package-/BIC/YEM_PF.
ZVAR12 = gw_result_package-/BIC/YEM_GRT.
ZVAR13 = gw_result_package-/BIC/YEM_PINC.
ZVAR14 = gw_result_package-/BIC/YEM_REMGR.

* gross salary = Basic Sal + DA + BOA
ZVAR15 = ZVAR8 + ZVAR9 + ZVAR10.

* gross salary = Basic Sal + DA + BOA + PF + GRT
ZVAR16 = ZVAR8 + ZVAR9 + ZVAR10 + ZVAR11 + ZVAR12.
* gross salary = Basic Sal + DA + BOA + PF + GRT + Perform. Incentive
ZVAR17 = ZVAR8 + ZVAR9 + ZVAR10 + ZVAR11 + ZVAR12 + ZVAR13.

* if reporting manager field is blank means he himself is the manager hence populate his own employee number else populate given manager employee number

If ZVAR14 EQ ' ':
   gw_result_package1-/BIC/YEM_REMGR = ZVAR1.
ELSE.
   gw_result_package1-/BIC/YEM_REMGR = ZVAR14.
ENDIF.

* storing the values & calculated values into internal package 1

   gw_result_package1-/BIC/YEM_NUM = ZVAR1.
gw_result_package1-/BIC/YEM_PH1 = ZVAR2.
gw_result_package1-/BIC/YEM_PH2 = ZVAR3.
gw_result_package1-/BIC/YEM_PH3 = ZVAR4.
gw_result_package1-/BIC/YEM_DSG = ZVAR5.
gw_result_package1-/BIC/YEM_JB = ZVAR6.
gw_result_package1-/BIC/YEM_CRR = ZVAR7.
gw_result_package1-/BIC/YEM_BSAL = ZVAR8.
gw_result_package1-/BIC/YEM_DA = ZVAR9.
gw_result_package1-/BIC/YEM_BOA = ZVAR10.
gw_result_package1-/BIC/YEM_PF = ZVAR11.
gw_result_package1-/BIC/YEM_GRT = ZVAR12.
gw_result_package1-/BIC/YEM_PINC = ZVAR13.
gw_result_package1-/BIC/YEM_GRS = ZVAR15.
gw_result_package1-/BIC/YEM_TGRS = ZVAR16.
gw_result_package1-/BIC/YEM_GRINC = ZVAR17.

* updating the result package by new values.

   append gw_RESULT_PACKAGE1 to RESULT_PACKAGE.

* clearing the local variables.

   CLEAR ZVAR1.
   CLEAR ZVAR2.
   CLEAR ZVAR3.
   CLEAR ZVAR4.
   CLEAR ZVAR5.
   CLEAR ZVAR6.
   CLEAR ZVAR7.
   CLEAR ZVAR8.
   CLEAR ZVAR9.
   CLEAR ZVAR10.
   CLEAR ZVAR11.
   CLEAR ZVAR12.
   CLEAR ZVAR13.
   CLEAR ZVAR14.
   CLEAR ZVAR15.
   CLEAR ZVAR16.
   CLEAR ZVAR17.

* End of the loop.
ENDLOOP.

* End of routine - insert your code only before this line

ENDMETHOD.

*----------------------------------------------------------------------*
*     Method inverse_expert_routine                                      *
*----------------------------------------------------------------------*
* This subroutine needs to be implemented only for direct access          *
* (for better performance) and for the Report/Report Interface          *
* (drill through).                                                       *
* The inverse routine should transform a projection and                  *
* a selection for the target to a projection and a selection             *
* for the source, respectively.                                          *
* If the implementation remains empty all fields are filled and           *
* all values are selected.                                              *
METHOD inverse_expert_routine.

"$ begin of inverse routine - insert your code only below this line."
... "Insert your code here
"$ end of inverse routine - insert your code only before this line "--"

ENDMETHOD.     "inverse_expert_routine
ENDCLASS.      "routine IMPLEMENTATION

Special Note:

If the target of the transformation is a DataStore object, key figures are updated by default with the aggregation behavior Overwrite (MOVE).

References:
1.  For more details refer to the link,
http://help.sap.com/saphelp_nw04s/helpdata/en/a4/1be541f321c717e10000000a155106/frameset.htm