

# Step by Step Creation of Custom Planning Function to Change Cube Data as Per Your Own Logic



## Applies to:

SAP BI 7.0. For more information, visit the [Business Intelligence Homepage](#).

## Summary

This article will explain the step by step procedure to retrieve cube data in an internal table via planning function exit. This document will also explain how cube data can be change via exit code. This document also shows how exit type planning function can be debugging via ABAP.

**Author:** Arup Goswami

**Company:** Fujitsu Consulting India Pvt. Ltd.

**Created on:** 25 September 2011.

## Author Bio



Arup Goswami has over 4 years consulting experience in SAP BI & IP projects and 3 years of ABAP/OO programming. Arup is presently working for Saudi Aramco, Saudi Arabia. He has previously worked in Capgemini and IBM.

## Table of Contents

|                                      |    |
|--------------------------------------|----|
| Scenario .....                       | 3  |
| Create Aggregation Level .....       | 3  |
| Exit Class Creation.....             | 5  |
| Planning Function Type Creation..... | 7  |
| Code Written in Exit Class .....     | 7  |
| Execute a Planning Sequence.....     | 11 |
| Check data in cube .....             | 13 |
| Debug a Planning Function Exit. .... | 14 |
| Related Content.....                 | 16 |
| Disclaimer and Liability Notice..... | 17 |

## Scenario

Let's take an example where cube data is like below mentioned way. It has loaded from flat file. Now the output for cube is bellow.

| YMEASURE1 | YMEA_VAL | ZSEMPLOORG | 0CHNGID | Record type | Request ID                    | 0FISCVARNT | Fiscal year | 0UNIT | Quantity |
|-----------|----------|------------|---------|-------------|-------------------------------|------------|-------------|-------|----------|
| M1        | A        | 500010     |         |             | DTPR_4N6CLNLI2NCAI3ELH2ROJ61Y | K4         | 2011        | KG    | 10.000   |
| M2        | A        | 500010     |         |             | DTPR_4N6CLNLI2NCAI3ELH2ROJ61Y | K4         | 2011        | KG    | 20.000   |
| M3        | A        | 500010     |         |             | DTPR_4N6CLNLI2NCAI3ELH2ROJ61Y | K4         | 2011        | KG    | 30.000   |
| M4        | A        | 500010     |         |             | DTPR_4N6CLNLI2NCAI3ELH2ROJ61Y | K4         | 2011        | KG    | 40.000   |
| M5        | A        | 500010     |         |             | DTPR_4N6CLNLI2NCAI3ELH2ROJ61Y | K4         | 2011        | KG    | 50.000   |
| M6        | A        | 500010     |         |             | DTPR_4N6CLNLI2NCAI3ELH2ROJ61Y | K4         | 2011        | KG    | 60.000   |
| M7        | A        | 500010     |         |             | DTPR_4N6CLNLI2NCAI3ELH2ROJ61Y | K4         | 2011        | KG    | 70.000   |

## Business Case

The user wants to create his own define planning function which will calculate as per his customs logic. In run time, he can take data in to ABAP internal table then he can calculate the data as per his own logic. So he needs to create an aggregation level, a filter, a customs planning function and a planning sequence.

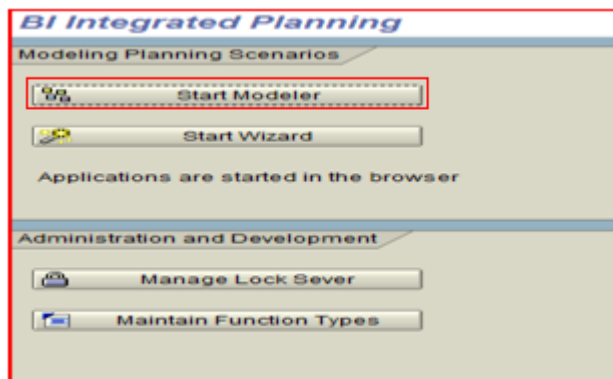
## Setting up the environment

Cube is defined like below mentioned way: It is a Real Time Cube. The cube should be in the position of planning mode.

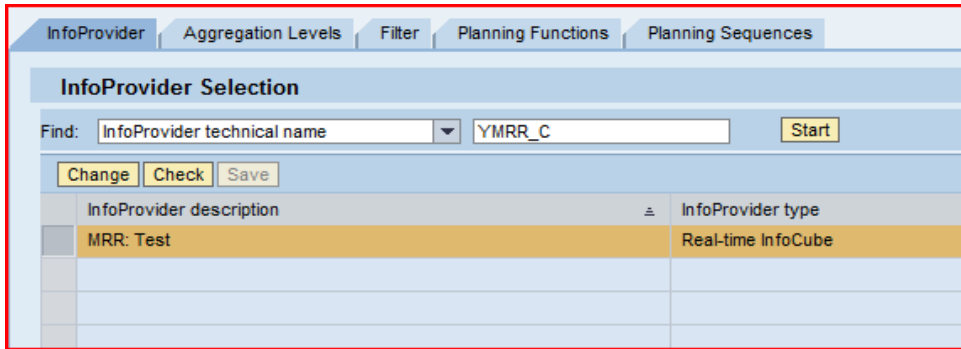
| Object Overview              | Field           | InfoObject | Data elem.        | Data Type | Length | DataSource |
|------------------------------|-----------------|------------|-------------------|-----------|--------|------------|
| Characteristics              |                 |            |                   |           |        |            |
| Measure / KPI                | /BIC/YMEASURE1  | YMEASURE1  | /BIC/OIYMEASURE1  | CHAR      | 20     |            |
| Measure Value                | /BIC/YMEA_VAL   | YMEA_VAL   | /BIC/OIYMEA_VAL   | CHAR      | 3      |            |
| Planning Organizational Unit | /BIC/ZSEMPLOORG | ZSEMPLOORG | /BIC/OIZSEMPLOORG | NUMC      | 8      |            |
| Key Figures                  |                 |            |                   |           |        |            |
| Quantity                     | QUANTITY        | 0QUANTITY  | /BI0/OIQUANTITY   | QUAN      | 9      |            |
| Time Characteristics         |                 |            |                   |           |        |            |
| Fiscal year                  | FISCYEAR        | 0FISCYEAR  | /BI0/OIFISCYEAR   | NUMC      | 4      |            |
| Fiscal year variant          | FISCVARNT       | 0FISCVARNT | /BI0/OIFISCVARNT  | CHAR      | 2      |            |
| Units                        |                 |            |                   |           |        |            |
| Unit of measure              | UNIT            | 0UNIT      | /BI0/OIUNIT       | UNIT      | 3      |            |
| Other Fields                 |                 |            |                   |           |        |            |
| Change Run ID                | CHNGID          | 0CHNGID    | /BI0/OICHNGID     | NUMC      | 14     |            |
| Record type                  | RECORDTP        | 0RECORDTP  | /BI0/OIRECORDTP   | NUMC      | 1      |            |
| Request ID                   | REQUID          | 0REQUID    | /BI0/OIREQUID     | CHAR      | 30     |            |

## Create Aggregation Level

Execute the tcode RSPLAN to start the planning object creation.

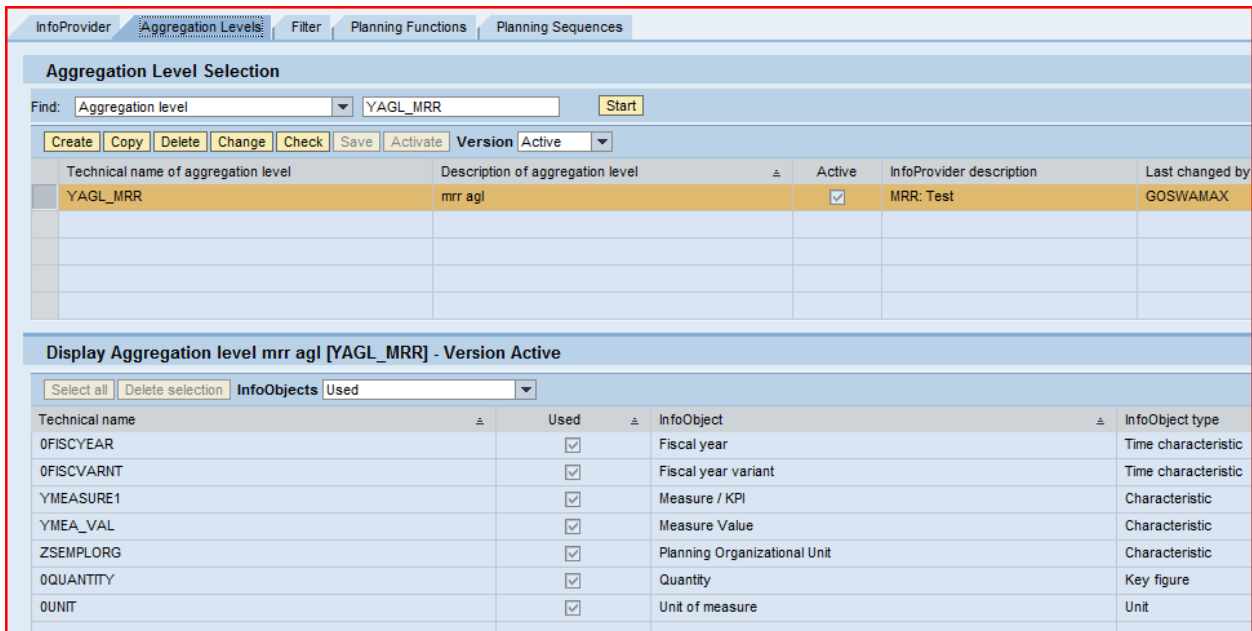


Execute the start modeler button to create planning objects. Aggregation level can be created on the top of info provider. So first you need to select the info provider name.



Here you need to provide the cube name as an info provider and press the Start button. For our example the cube name YMRR\_C.

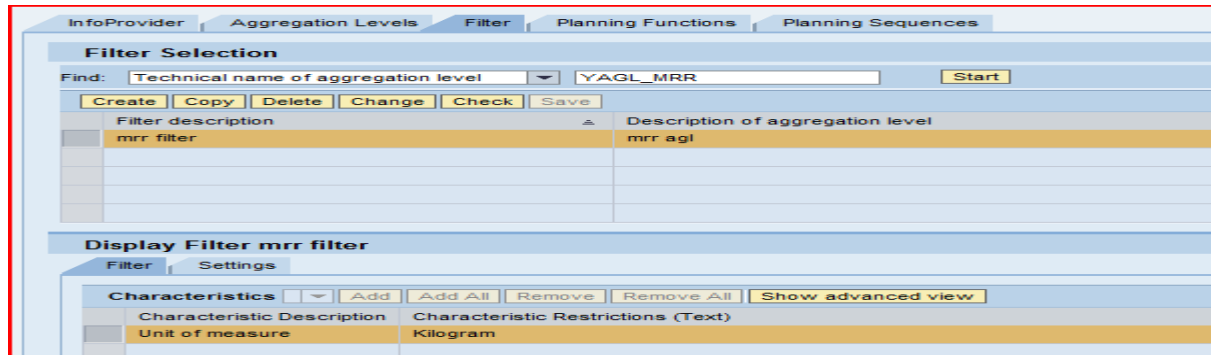
Go to Aggregation Level tab and put a name in YAGL\_MRR. Select all the info objects in the used column.



Save and activate the aggregation level.

## Create Filter

Go to the filter tab and create a filter for our planning function. For our example, we have used unit of measure as 'KG'. Give a technical name for filter and Save it.

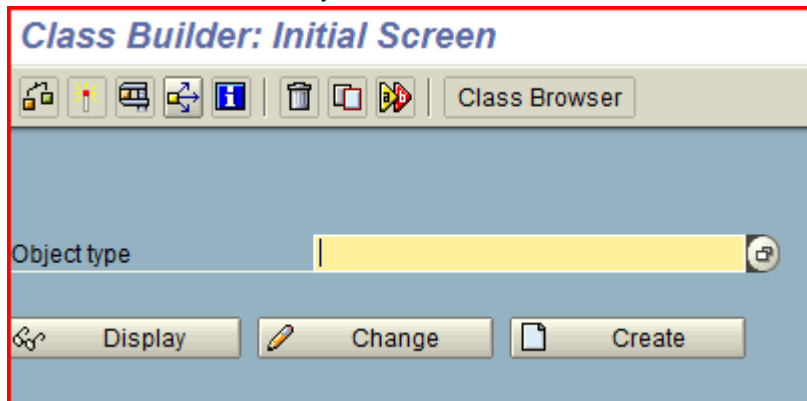


## Create Planning Function

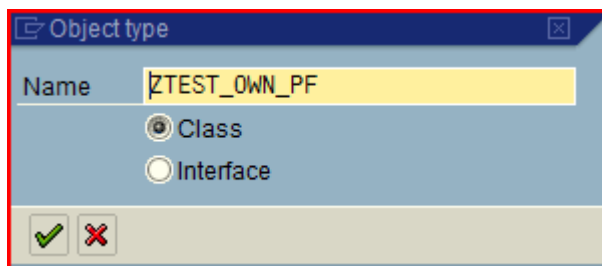
To create my own planning function we need to perform few steps.

### Exit Class Creation

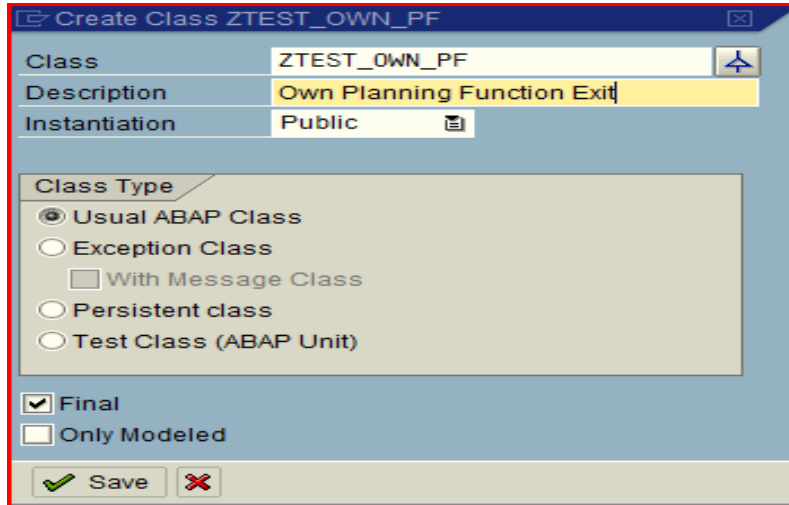
- (a) Go to tcode SE24. Create your own class ZTEST\_PF\_TYP.



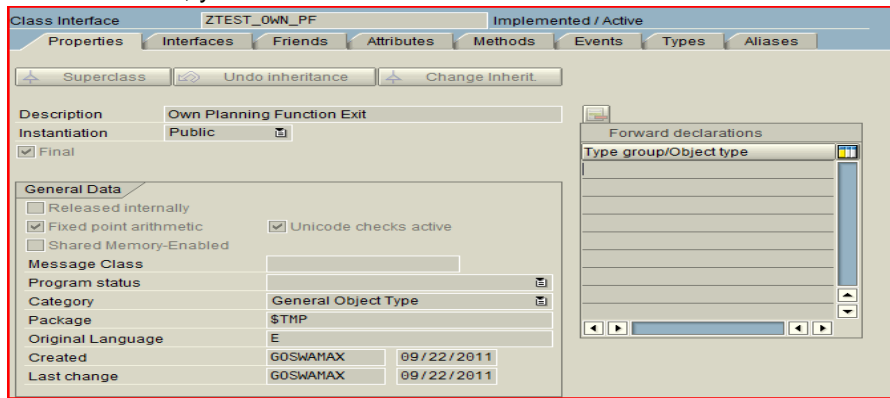
- (b) Select the class radio button.



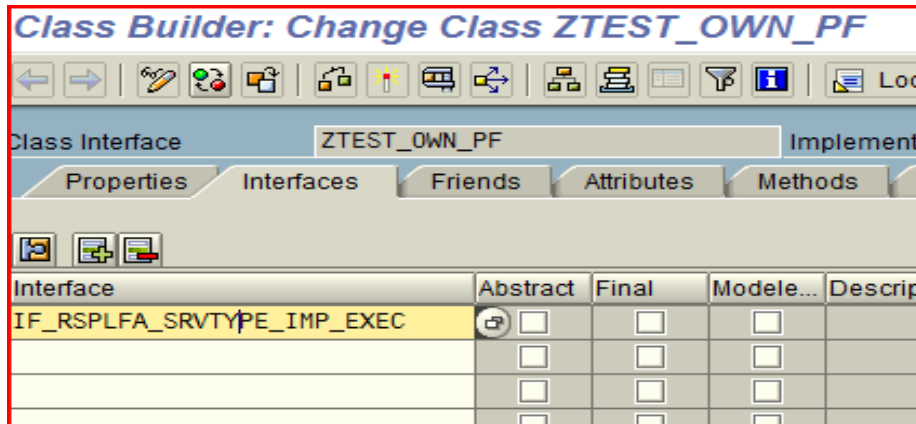
(c) Provide the Description and Save it. Activate it also.



After activation, you can find the below screen.

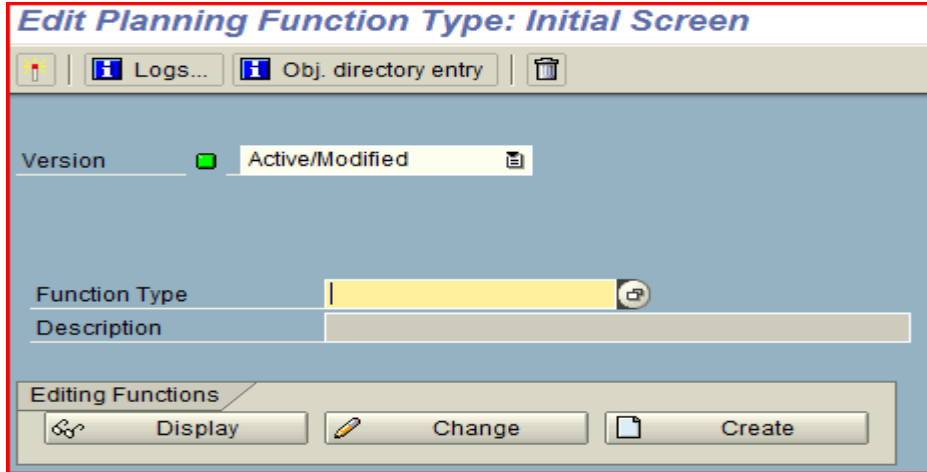


(d) Put **IF\_RSPLFA\_SRVTYPE\_IMP\_EXEC** in interface tab. Save and activate it.

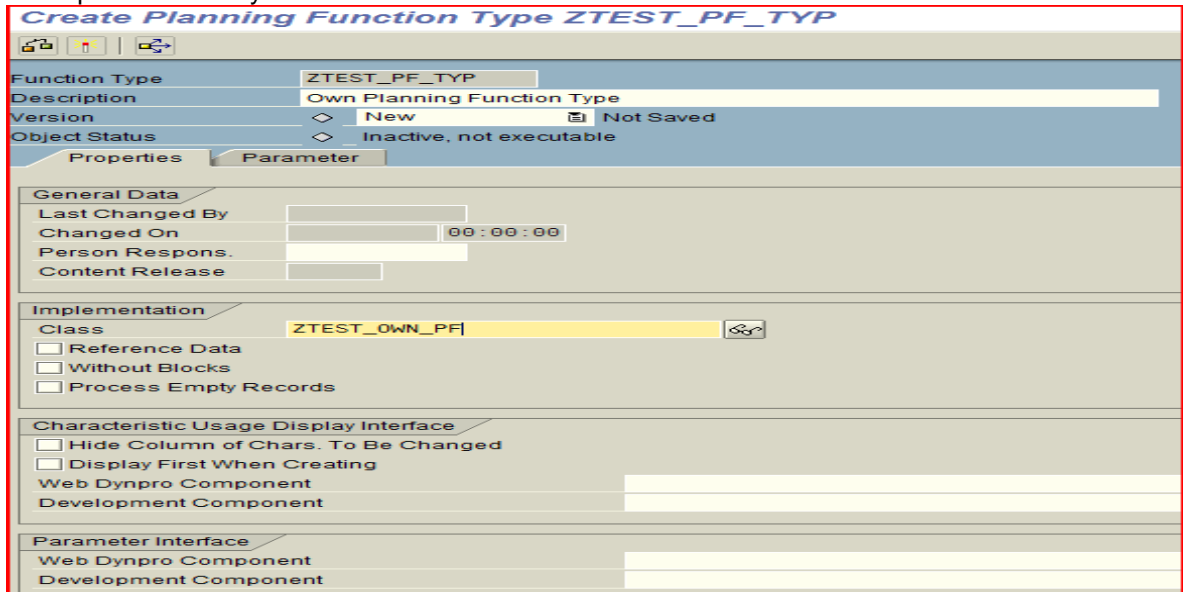


### Planning Function Type Creation

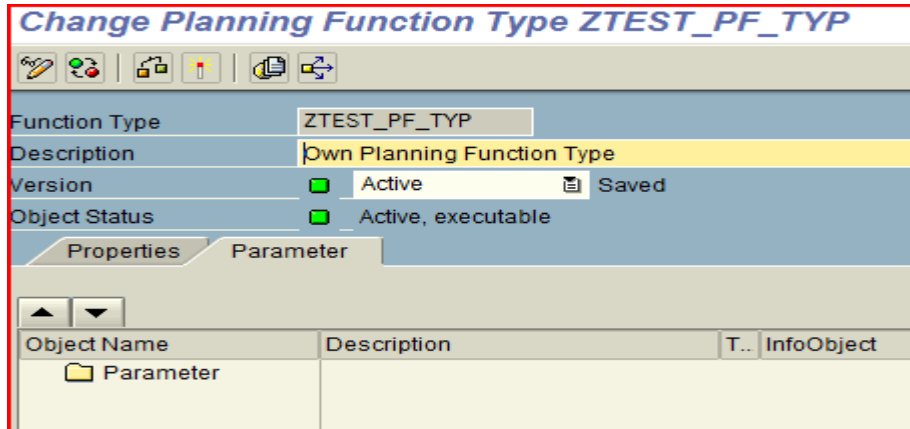
- (a) Go to tcode **RSPLF1** to create planning function type.



Put Description and newly created ABAP Class. Activate it.



For parameter tab nothing is required for our examples.

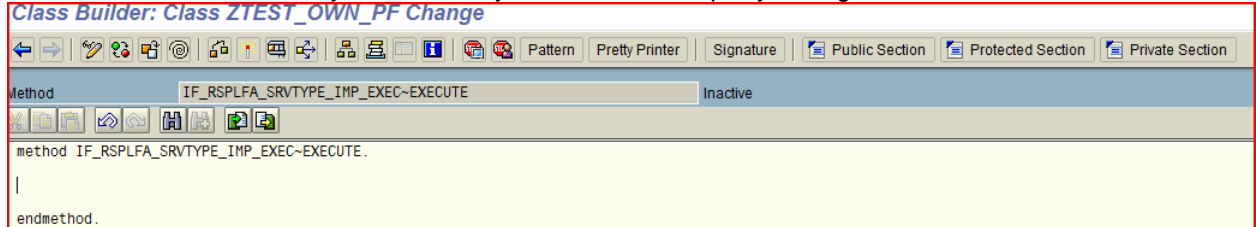


### Code Written in Exit Class

- (a) Go to **Method** tab in exit class.

| Class Interface   |            | ZTEST_OWN_PF |            |                              | Implemented / Inactive |       |         |
|---|------------|--------------|------------|------------------------------|------------------------|-------|---------|
| Properties  | Interfaces | Friends      | Attributes | Methods                      | Events                 | Types | Aliases |
| <input type="checkbox"/> Parameters <input checked="" type="checkbox"/> Exceptions <input type="checkbox"/> <input type="checkbox"/> Filter |            |              |            |                              |                        |       |         |
| Method  | Level      | Visi...      | M...       | Description                  |                        |       |         |
| IF_RSPLFA_SRVTYPE_IMP_EXEC~INIT_EXECUTION   | Insta...   | Pub...       |            | Initialization for Execution |                        |       |         |
| IF_RSPLFA_SRVTYPE_IMP_EXEC~EXECUTE  | Insta...   | Pub...       |            | Execution                    |                        |       |         |
| IF_RSPLFA_SRVTYPE_IMP_EXEC~FINISH_EXECUTION   | Insta...   | Pub...       |            | Actions at End of Execution  |                        |       |         |

Double click on **Execute**. Here you can write your own code as per your logic.



(b) To collect the data in a container, you need to create an internal table. Go to **Public Section** and create your own container type. In our example, we have created a work area type and a table type.

**\*\*>> Work area for Internal Table (cube)**

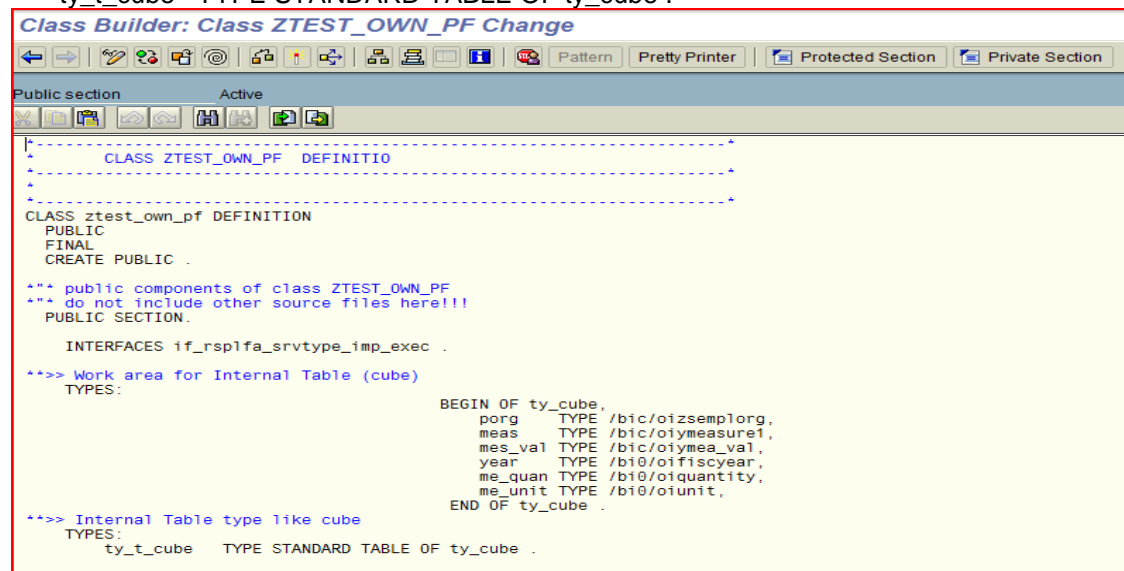
TYPES:

```
BEGIN OF ty_cube,
  porg TYPE /bic/oiemplorg,
  meas TYPE /bic/oiymeasure1,
  mes_val TYPE /bic/oiymea_val,
  year TYPE /bi0/oifiscyear,
  me_quan TYPE /bi0/oiquantity,
  me_unit TYPE /bi0/oiunit,
END OF ty_cube .
```

**\*\*>> Internal Table type like cube**

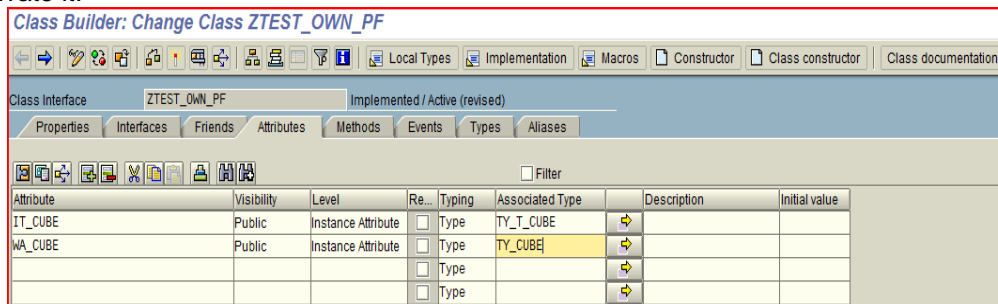
TYPES:

```
ty_t_cube TYPE STANDARD TABLE OF ty_cube .
```





Go to **Attribute** tab in Class to define internal table and work area globally. Put the global work area name wa\_cube (type ty\_cube) and global internal table name it\_cube (type ty\_t\_cube). Save and Activate it.



(c) Write the below mentioned code to store the data in internal table.

```
DATA : l_r_s_data      TYPE REF TO data,
       l_data_tab      TYPE REF TO data.
```

```
FIELD-SYMBOLS: <data_tab>    TYPE ANY TABLE,
               <l_s_data>     TYPE ANY,
               <p_org>        TYPE ANY,
               <measure>     TYPE ANY,
               <me_val>      TYPE ANY,
               <me_quan>     TYPE ANY,
               <fis_year>    TYPE ANY,
               <me_unit>     TYPE ANY.
```

```
CREATE DATA l_data_tab    LIKE STANDARD TABLE OF c_th_data INITIAL SIZE 0.
ASSIGN l_data_tab->* TO <data_tab>.
CREATE DATA l_r_s_data    LIKE LINE OF c_th_data.
**>>We need change C_TH_DATA for cube data change.
ASSIGN l_r_s_data->*      TO <l_s_data>.
```

```
**>> Retrieve all value from cube into internal table.
LOOP AT c_th_data INTO <l_s_data>.
```

```
ASSIGN COMPONENT 'ZSEMPLOG' OF STRUCTURE <l_s_data> TO <p_org>. " Org
ASSIGN COMPONENT 'YMEASURE1' OF STRUCTURE <l_s_data> TO <measure>. " Measure
ASSIGN COMPONENT 'YMEA_VAL' OF STRUCTURE <l_s_data> TO <me_val>. " Meas Val
ASSIGN COMPONENT '0FISCYEAR' OF STRUCTURE <l_s_data> TO <fis_year>. " Year
ASSIGN COMPONENT '0QUANTITY' OF STRUCTURE <l_s_data> TO <me_quan>. " Qty
ASSIGN COMPONENT '0UNIT' OF STRUCTURE <l_s_data> TO <me_unit>. " Unit
```

```
**>> We have not given Fiscal Year Variant because we have fixed it 'K4' in cube level.
CLEAR wa_cube.
wa_cube-porg = <p_org>.
wa_cube-meas = <measure>.
wa_cube-mes_val = <me_val>.
wa_cube-year = <fis_year>.
wa_cube-me_quan = <me_quan>.
wa_cube-me_unit = <me_unit>.
```

```
**>> Retrieve the cube data in to internal table.
COLLECT wa_cube INTO it_cube[] .
```

```
CLEAR wa_cube.
ENDLOOP.
```

Create Planning Function from RSPLAN >Start Modeler>Planning Function Tab.

**Create Planning Function**

Type: Own Planning Function Type

Technical Name: \* yown\_typ Description: Own Plannign Function

**Aggregation Level Selection**

Find: Technical name of aggregation level YAGL\_MRR Start Help

Filter on Settings

| Technical name of aggregation level | Description of aggregation level | Active                              | InfoProvider description | Last changed by | Date      | Time       |
|-------------------------------------|----------------------------------|-------------------------------------|--------------------------|-----------------|-----------|------------|
| YAGL_MRR                            | mrr agl                          | <input checked="" type="checkbox"/> | MRR: Test                | GOSWAMAX        | 9/16/2011 | 2:48:47 PM |

Transfer Cancel

Save and activate the planning function.

Planning function Own Plannign Function (YOWN\_TYP) is consistent

InfoProvider Aggregation Levels Filter Planning Functions Planning Sequences

**Planning Function Selection**

Find: Technical name of aggregation level YAGL\_MRR Start Help

Create Copy Delete Display Check Save Filter on Settings

| Planning function description | Type                           | Aggregation level description | Last changed by | Date      | Time       |
|-------------------------------|--------------------------------|-------------------------------|-----------------|-----------|------------|
| MRR 1                         | Measure Result Recalculation 1 | mrr agl                       | GOSWAMAX        | 9/21/2011 | 3:54:21 PM |
| Own Plannign Function         | Own Planning Function Type     | mrr agl                       | GOSWAMAX        | 9/22/2011 | 5:42:26 PM |
| mrr pf                        | Measure Result Recalculation   | mrr agl                       | GOSWAMAX        | 9/16/2011 | 2:49:28 PM |

**Change Planning function Own Plannign Function - Characteristic usage**

To Parameters To Characteristic Usage

- The values of which characteristics are to be changed when the required planning function is executed on a record or block?

**Characteristic**

Select all Select no characteristics Filter on Settings

| Characteristic               | changed                  | used in conditions       |
|------------------------------|--------------------------|--------------------------|
| Fiscal year                  | <input type="checkbox"/> | <input type="checkbox"/> |
| Unit of measure              | <input type="checkbox"/> | <input type="checkbox"/> |
| Measure / KPI                | <input type="checkbox"/> | <input type="checkbox"/> |
| Measure Value                | <input type="checkbox"/> | <input type="checkbox"/> |
| Planning Organizational Unit | <input type="checkbox"/> | <input type="checkbox"/> |

### Create Planning Sequence

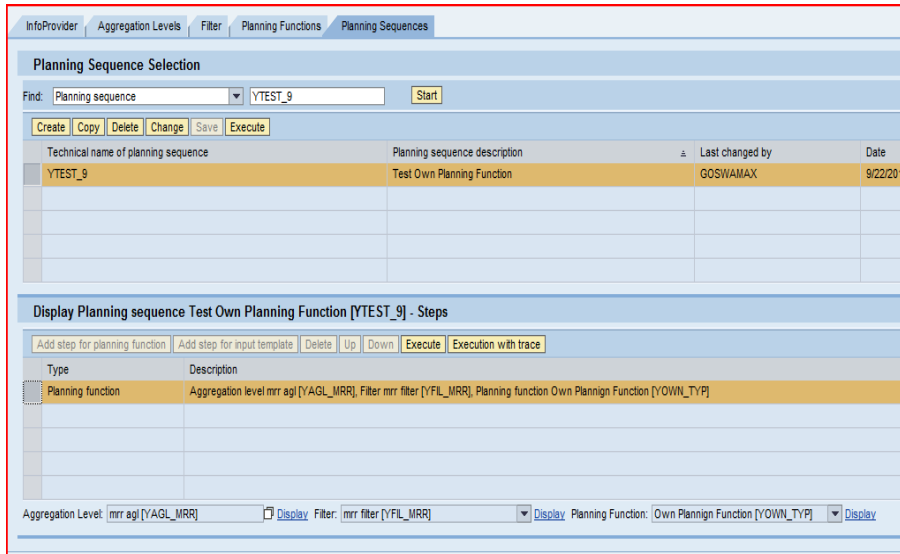
Create Planning Sequence.

**Create Planning Sequence**

Technical Name: \* ytest\_9 Description: Test Own Planning Function

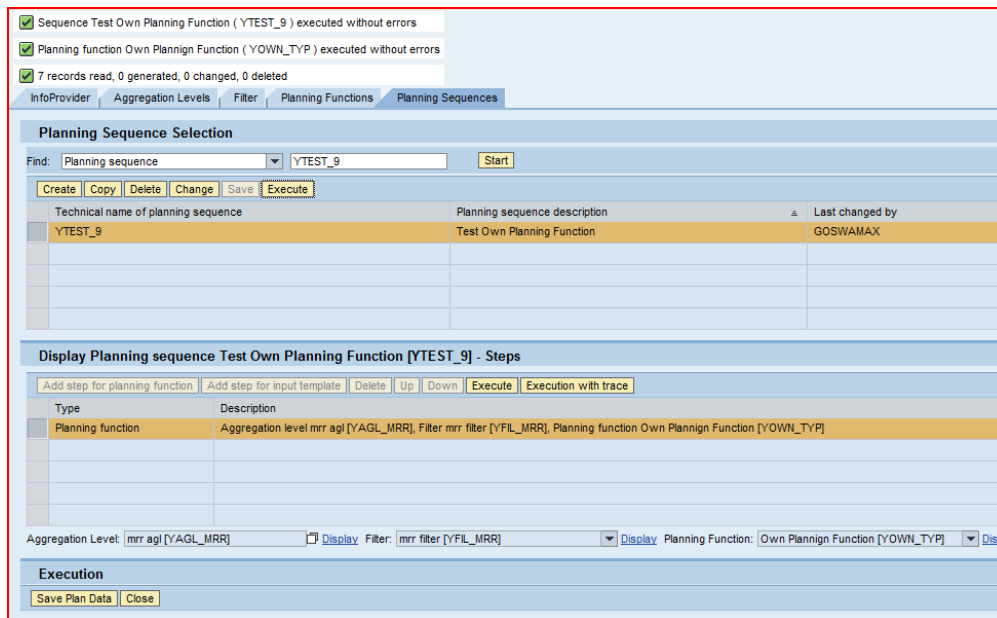
Transfer Cancel

Save it. Add Filter and Planning Function name in the planning sequence.



## Execute a Planning Sequence

If you execute the planning sequence then you will find the below mention screen shot. You can find easily that 7 records have been read by planning sequence.



In exit code, still now we have not written any code where we have change the C\_TH\_DATA. So we are getting only 7 records have been read. Now we are just adding in every record '10' extra. So 7 records have been read and 7 records have change.

**Exit code has been changed:**

```
DATA :      l_r_s_data      TYPE REF TO data,
          l_data_tab       TYPE REF TO data.
```

```
FIELD-SYMBOLS: <data_tab>  TYPE ANY TABLE,
                <l_s_data>  TYPE ANY,
                <p_org>     TYPE ANY,
                <measure>   TYPE ANY,
                <me_val>    TYPE ANY,
                <me_quan>   TYPE ANY,
                <fis_year>  TYPE ANY,
                <me_unit>   TYPE ANY.
```

```
CREATE DATA l_data_tab LIKE STANDARD TABLE OF c_th_data INITIAL SIZE 0.
```

```
ASSIGN l_data_tab->* TO <data_tab>.
```

```
CREATE DATA l_r_s_data LIKE LINE OF c_th_data.
```

```
ASSIGN l_r_s_data->* TO <l_s_data>.
```

**\*\*>> Retrieve all value from cube into internal table.**

```
LOOP AT c_th_data INTO <l_s_data>.
```

```
  ASSIGN COMPONENT 'ZSEMPLORG' OF STRUCTURE <l_s_data> TO <p_org>. " Org
  ASSIGN COMPONENT 'YMEASURE1' OF STRUCTURE <l_s_data> TO <measure>. " Measure
  ASSIGN COMPONENT 'YMEA_VAL' OF STRUCTURE <l_s_data> TO <me_val>. " Measure Value
  ASSIGN COMPONENT '0FISCYEAR' OF STRUCTURE <l_s_data> TO <fis_year>. " Year
  ASSIGN COMPONENT '0QUANTITY' OF STRUCTURE <l_s_data> TO <me_quan>. " Quantity
  ASSIGN COMPONENT '0UNIT' OF STRUCTURE <l_s_data> TO <me_unit>. " Unit
```

**\*\*>> Add ten (10) extra in quantity.**

```
  <me_quan> = <me_quan> + '10'.
```

**\*\*>> C\_TH\_DATA modifying that means it will save the data in cube.**

```
  MODIFY TABLE c_th_data FROM <l_s_data>.
```

```
ENDLOOP.
```

**Output from planning Sequence run:**

The screenshot displays the SAP Planning Sequences interface. At the top, three green status messages indicate successful execution: 'Sequence Test Own Planning Function ( YTEST\_9 ) executed without errors', 'Planning function Own Plannign Function ( YOWN\_TYP ) executed without errors', and '7 records read, 0 generated, 7 changed, 0 deleted'. Below this, the 'Planning Sequence Selection' section shows 'YTEST\_9' selected. A table lists the sequence details: 'Test Own Planning Function' by 'GOSWAMAX' on '9/22/2011'. The 'Display Planning sequence Test Own Planning Function [YTEST\_9] - Steps' section shows a single step: 'Aggregation level mrr agl [YAGL\_MRR], Filter mrr filter [YFIL\_MRR], Planning function Own Plannign Function [YOWN\_TYP]'. At the bottom, the 'Execution' section shows the current configuration: 'Aggregation Level: mrr agl [YAGL\_MRR]', 'Filter: mrr filter [YFIL\_MRR]', and 'Planning Function: Own Plannign Function [YOWN\_TYP]'.

## Check data in cube

If you save the data then it will affect in cube data.

A request has been loaded in to cube.



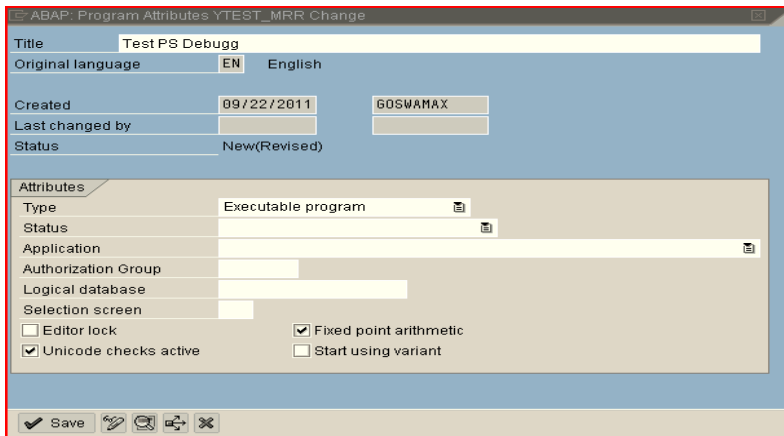
Change in cube data.

| YMEASURE1 | YMEA_VAL | ZSEMPLO | ORG | 0CHNGID | Record type | Request ID                     | 0FISCVARNT | Fiscal year | 0UNIT | Quantity |
|-----------|----------|---------|-----|---------|-------------|--------------------------------|------------|-------------|-------|----------|
| M1        | A        | 500010  |     |         |             | APO_R4N7YXXZ07M0B9IOWL06W3NDYE | K4         | 2011        | KG    | 10.000   |
| M2        | A        | 500010  |     |         |             | APO_R4N7YXXZ07M0B9IOWL06W3NDYE | K4         | 2011        | KG    | 10.000   |
| M3        | A        | 500010  |     |         |             | APO_R4N7YXXZ07M0B9IOWL06W3NDYE | K4         | 2011        | KG    | 10.000   |
| M4        | A        | 500010  |     |         |             | APO_R4N7YXXZ07M0B9IOWL06W3NDYE | K4         | 2011        | KG    | 10.000   |
| M5        | A        | 500010  |     |         |             | APO_R4N7YXXZ07M0B9IOWL06W3NDYE | K4         | 2011        | KG    | 10.000   |
| M6        | A        | 500010  |     |         |             | APO_R4N7YXXZ07M0B9IOWL06W3NDYE | K4         | 2011        | KG    | 10.000   |
| M7        | A        | 500010  |     |         |             | APO_R4N7YXXZ07M0B9IOWL06W3NDYE | K4         | 2011        | KG    | 10.000   |
| M1        | A        | 500010  |     |         |             | DTPR_4N6CLNLI2NCAI3ELH2ROJ61Y  | K4         | 2011        | KG    | 10.000   |
| M2        | A        | 500010  |     |         |             | DTPR_4N6CLNLI2NCAI3ELH2ROJ61Y  | K4         | 2011        | KG    | 20.000   |
| M3        | A        | 500010  |     |         |             | DTPR_4N6CLNLI2NCAI3ELH2ROJ61Y  | K4         | 2011        | KG    | 30.000   |
| M4        | A        | 500010  |     |         |             | DTPR_4N6CLNLI2NCAI3ELH2ROJ61Y  | K4         | 2011        | KG    | 40.000   |
| M5        | A        | 500010  |     |         |             | DTPR_4N6CLNLI2NCAI3ELH2ROJ61Y  | K4         | 2011        | KG    | 50.000   |
| M6        | A        | 500010  |     |         |             | DTPR_4N6CLNLI2NCAI3ELH2ROJ61Y  | K4         | 2011        | KG    | 60.000   |
| M7        | A        | 500010  |     |         |             | DTPR_4N6CLNLI2NCAI3ELH2ROJ61Y  | K4         | 2011        | KG    | 70.000   |

## Debug a Planning Function Exit.

- (a) Create a Test ABAP Program.

Go to tcode SE38 to create an ABAP program. Put a name and activate it.



Write a code like below mention way. Call the planning sequence **YTEST\_9** via function module from ABAP program.

DATA: e\_tk\_return LIKE bapiret2 OCCURS 0.

CALL FUNCTION 'RSPLSSE\_PLSEQ\_EXECUTE'

EXPORTING

i\_seqnm = 'YTEST\_9'

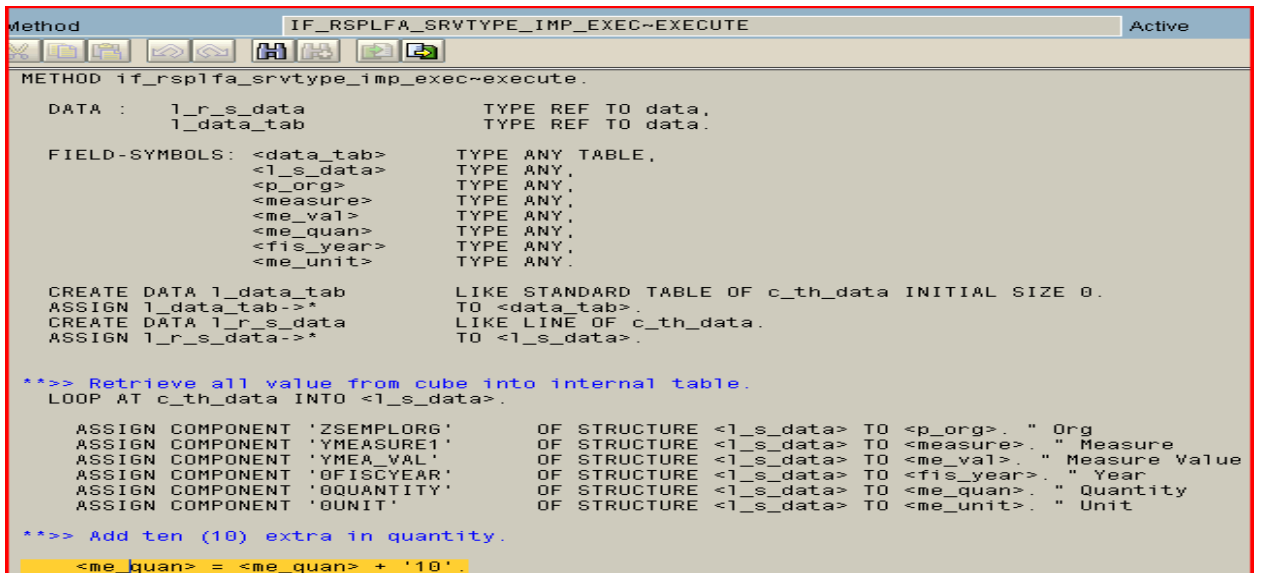
\* I\_VARIANT = ''

\* I\_FAST\_ENQUEUE = RS\_C\_FALSE

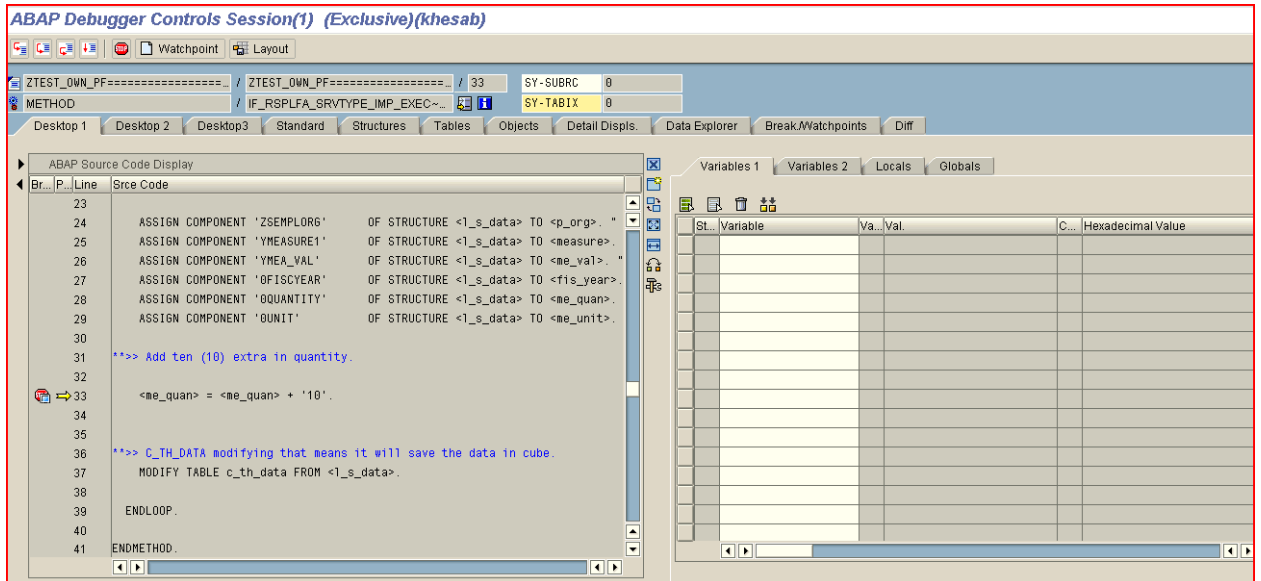
TABLES

e\_tk\_return = e\_tk\_return

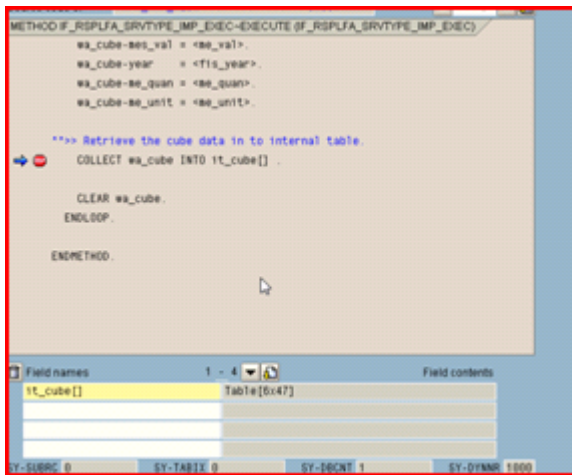
- (b) Set Break point in Exit Code. Go to tcode SE24 and put a break point.



- (c) Press F8 button to run ABAP Program from SE38. It will start the debug screen.



You can easily check how many records are present in internal table.



Hare we can see at this point we can 6 records are present in internal table.

## Related Content

<http://www.sdn.sap.com/irj/scn/index?rid=/library/uuid/90d200f5-c832-2e10-02ac-ac23f6544eb4>

<http://www.sdn.sap.com/irj/sdn/go/portal/prtroot/docs/library/uuid/90cef248-29e0-2910-11bb-df8dece7e22c>



## **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.