**BAPI Customized Steps for Customer Master # 01**

***Program theme:*** *In this demo program you can learn how to create the customized BAPI (Business Application Programming Interface) for Customer general master data and company code data.*

**5 Main Steps are involved in this:**

**Step-1:** Create Structure (T-Code SE11).

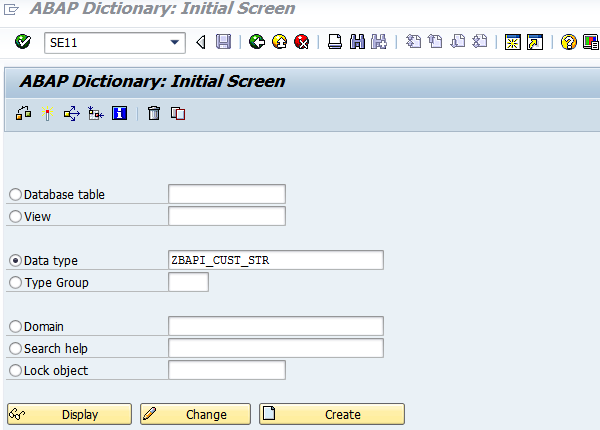
**Step-I1:** Create RFC Function Module (T-Code SE37).

**Step-II1:** Create Business Object (T-Code SWO1).

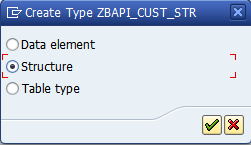
**Step-1V:** Check the BAPI Explorer (T-Code BAPI).

**Step-V:** Test BAPI Business Object (T-Code SWO1).

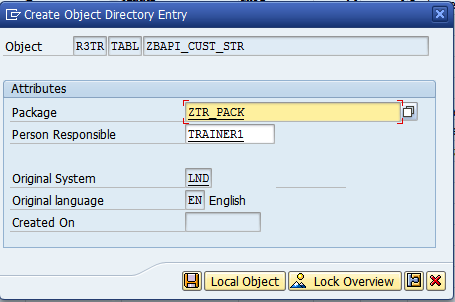
**Step # 1:** Go to SE11 T-Code and Select the Data type and provide the Structure name and click on create.



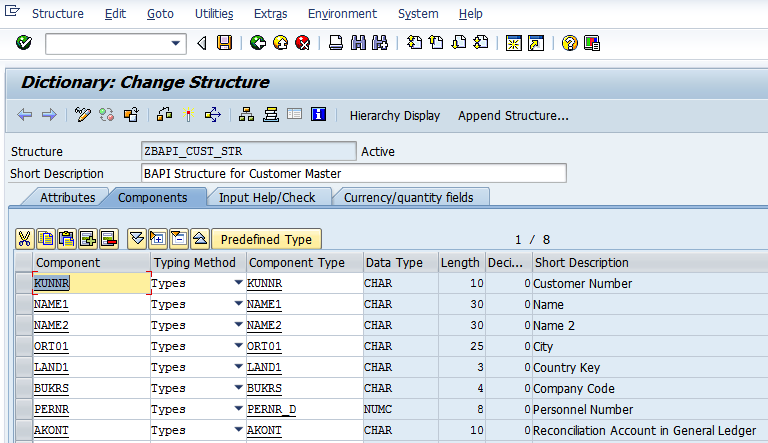
**Step # 2:** Select the Structure radio button and click on continue.



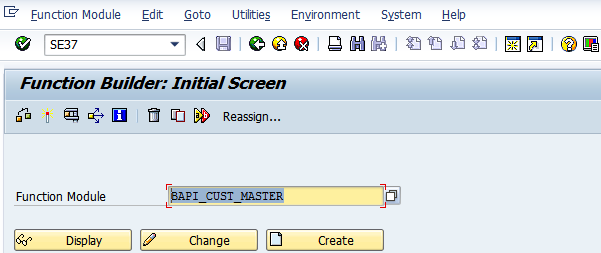
**Step # 3:** Save it in a package.



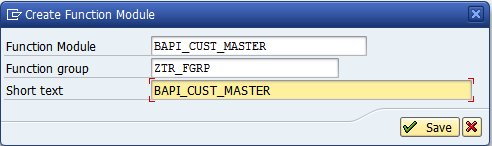
**Step # 4:** Provide the structure short description and copy the fields from KNA1 and KNB1 tables. Save and activate the Structure.



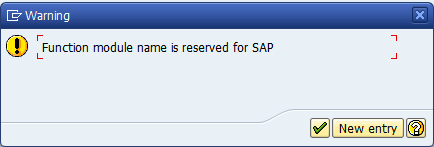
**Step # 5:** Go to SE37 T-Code and provide the function module name (Should Start with “BAPI\_ “ ) and click on create.



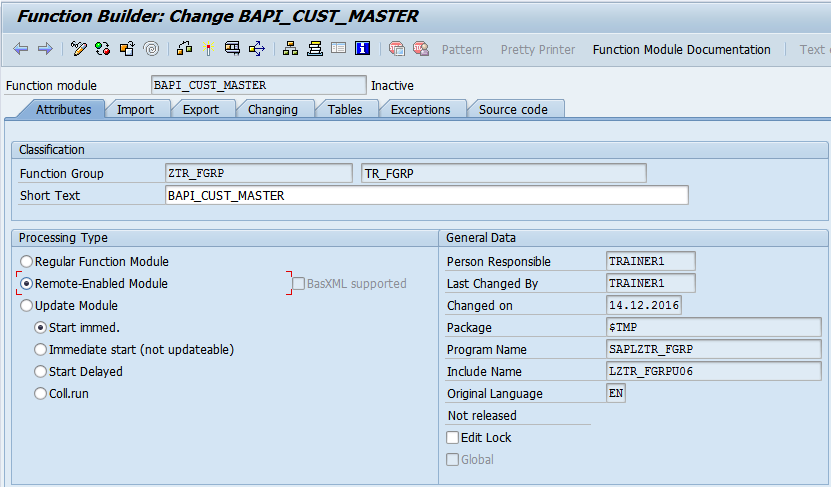
**Step # 6:** Provide the Function group name and short text for the function module and click on Save.



**Step # 7:** click on continue in warning window.

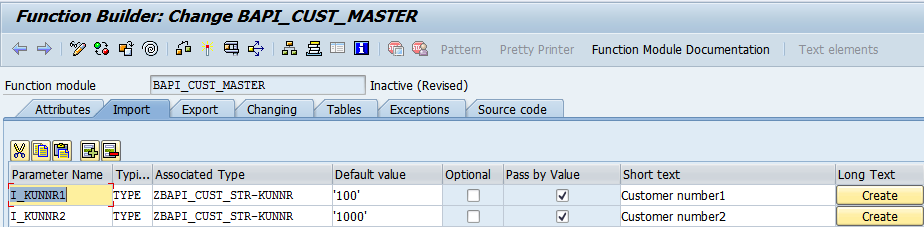


**Step # 8:** Provide the short text and go to **Attributes tab** and select the Remote-Enabled Module (RFC) radio button.



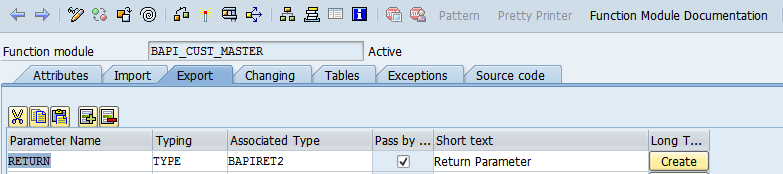
**Step # 9:** Go to **Import tab** and Provide the Parameter names and its associated type select from the Step-1 Structure-filed names.

Note: Remote-Enabled Module of Parameters, check box should be **pass by value** only; it cannot be default by **reference parameters**.



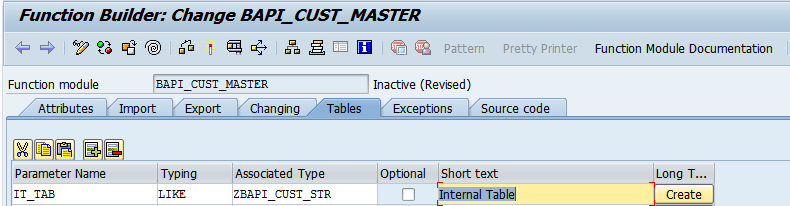
**Step # 10:** Go to **Export tab** and provide the parameter name and its associated type is predefined BAPIRET2.

Note: Export parameter RETURN check box should be **pass by value** only; it cannot be default by **reference parameters**.

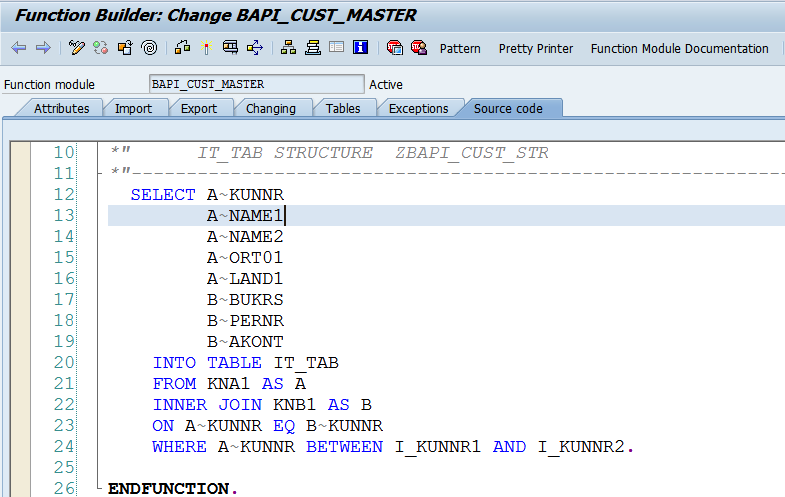


**Step # 11:** Go to **Tables tab** and provide the internal table name and its associated type as Structure name (From the Step-1).

FYI: you will get warning message then click on Enter button in the keyboard.

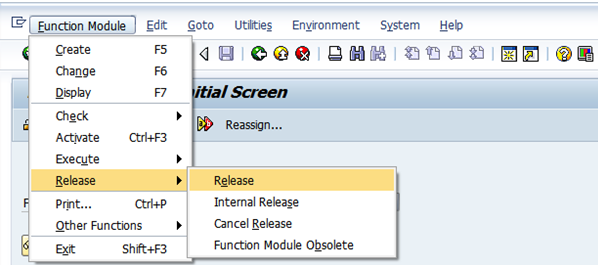


**Step # 12:** Go to Source code tab and write the program code for the function module.



**Step # 13:** Save and Activate the Function Module. Test function Module. Click on Back (F3). In the initial function module screen go to Function Module 🡪 Release 🡪 click on Release.

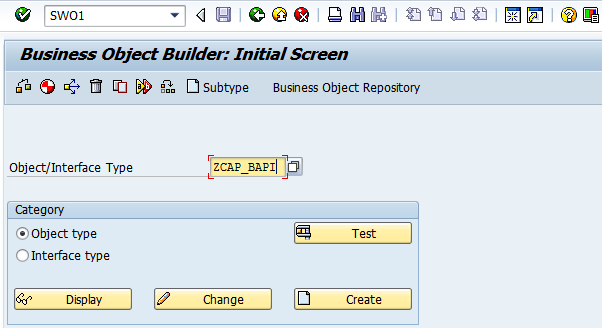
When you release it is available for RFC and BAPI.



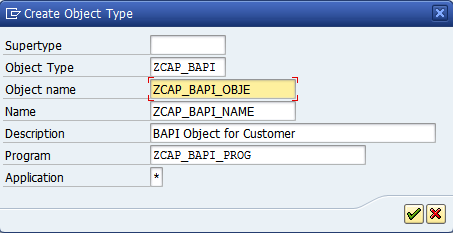
**Step # 14:** you will get the below message in the status bar.



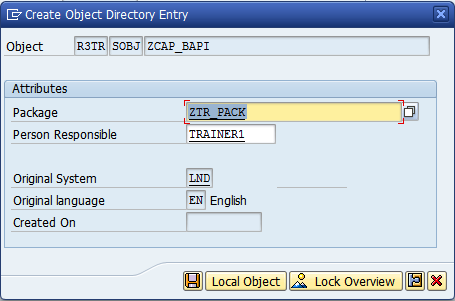
**Step # 15:** Go to SWO1 (“O - it’s not zero” ) T-Code and provide the Object name and click on create.



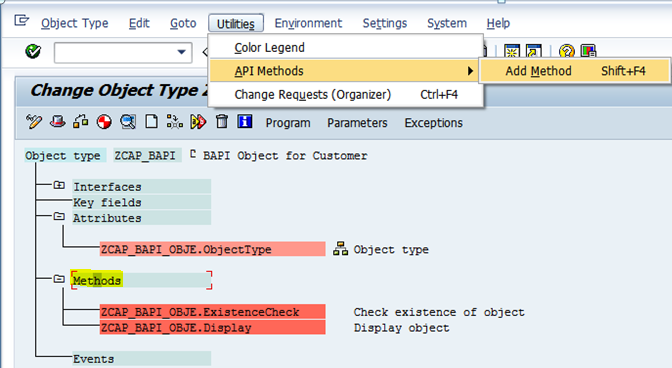
**Step # 16:** Provide the Object Type, Object name, Name, Description, Program and Application (\* Cross-Application) and click on continue.



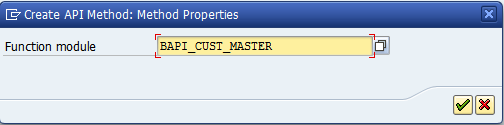
**Step # 17:** Save it in a package.



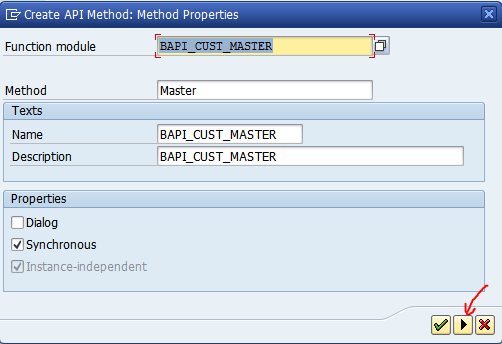
**Step # 18:** Place the cursor on the Methods and go to Utilities and Select the API Methods and Click Add Method to add the Function module as a method.



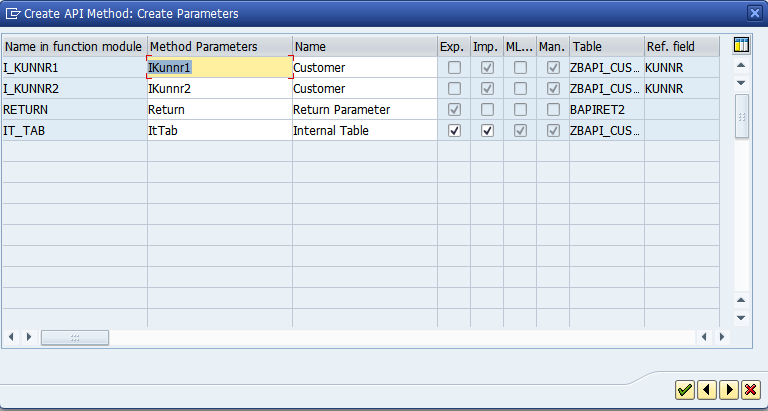
**Step # 19:** Enter the Function module name (user defined) and click on continue.



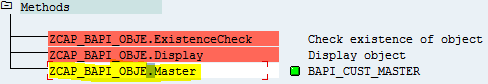
**Step # 20:** Click on Next Step icon (Arrow icon).



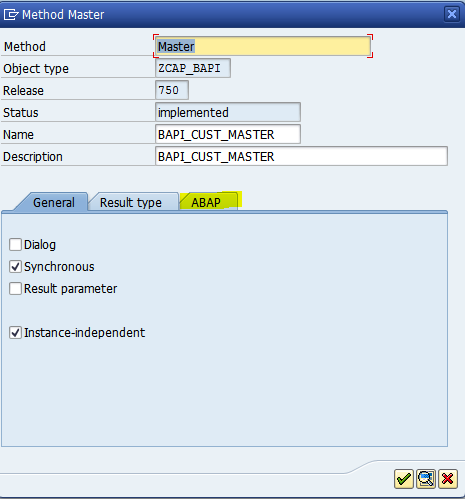
**Step # 21:** Click on continue.



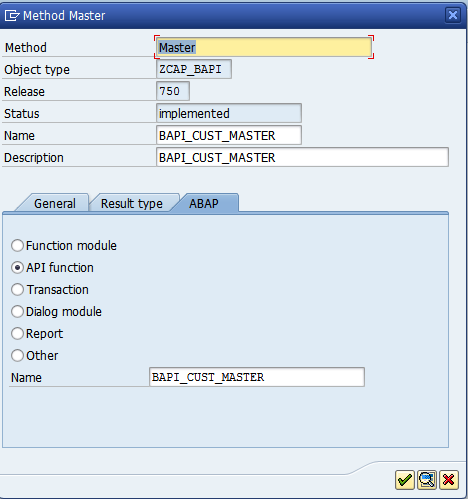
**Step # 22:** Place the cursors on the user defined method and double click on it.



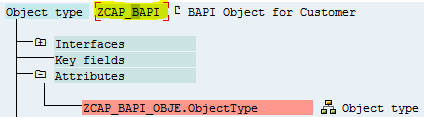
**Step # 23:** Select the ABAP Tab.

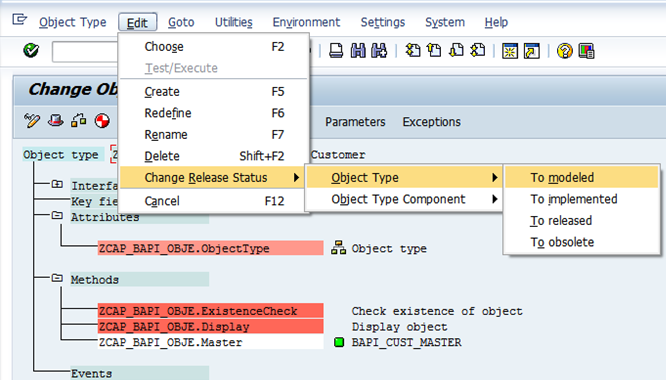


**Step # 24:** Select the API function radio button and click on continue.

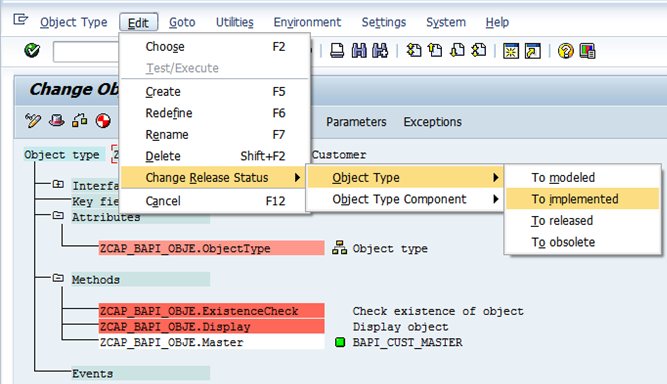


**Step # 25:** Select the Object type I.e ZCAP\_BAPI.

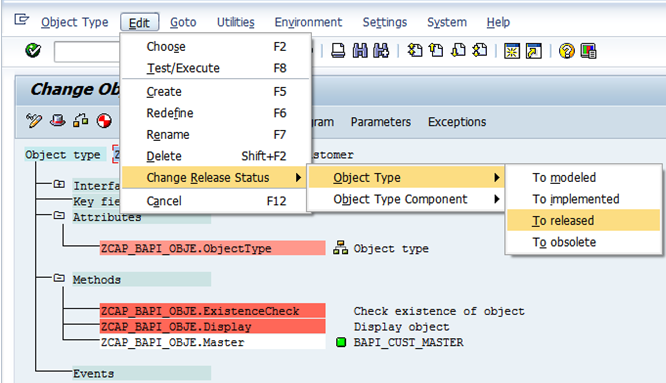


**Step # 26:** Place the cursor on Object type (I.e ZCAP\_BAPI) and go to Edit 🡪 Change Release Status 🡪 Object Type and select the To modeled.

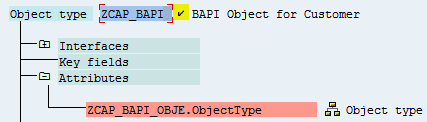
**Step # 27:** Place the cursor on Object type (I.e ZCAP\_BAPI) and go to Edit 🡪 Change Release Status 🡪 Object Type and select the To implemented.



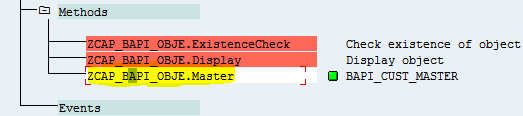
**Step # 28:** Place the cursor on Object type (I.e ZCAP\_BAPI) and go to Edit 🡪 Change Release Status 🡪 Object Type and select the To released.



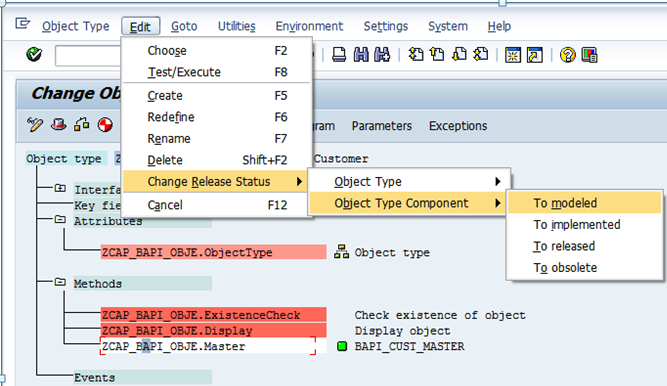
**Step # 29:** Once the object is implemented successfully you will get the tick symbol.



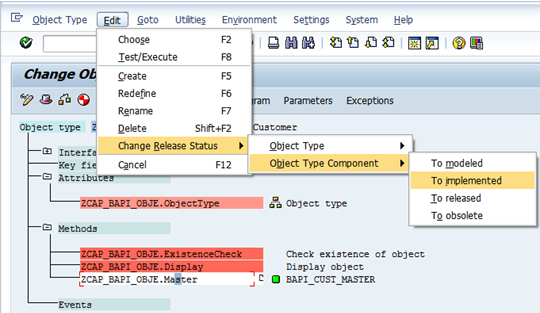
**Step # 30:** Select the Method I.e ZCAP\_BAPI\_OBJE.



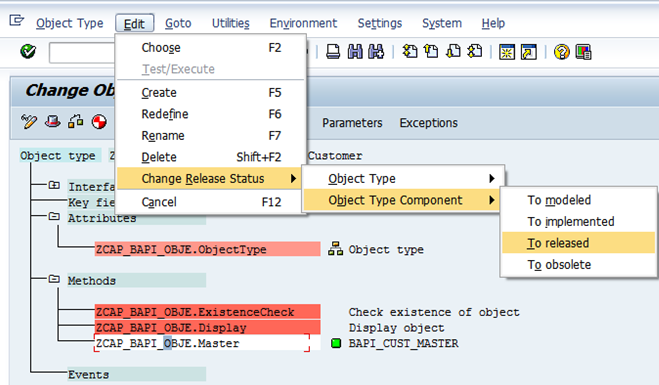
**Step # 31:** Place the cursor on Method (I.e ZCAP\_BAPI\_OBJE) and go to Edit 🡪 Change Release Status 🡪 Object Type Component and select the To modeled.



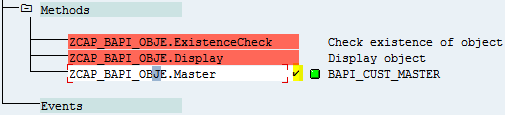
**Step # 32:** Place the cursor on Method (I.e ZCAP\_BAPI\_OBJE) and go to Edit 🡪 Change Release Status 🡪 Object Type Component and select the To implemented.



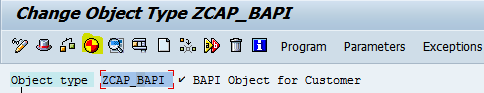
**Step # 33:** Place the cursor on Method (I.e ZCAP\_BAPI\_OBJE) and go to Edit 🡪 Change Release Status 🡪 Object Type Component and select the To released.



**Step # 34:** Once the object is implemented successfully you will get the tick symbol.



**Step # 35:** Select the object type and click on generate button.



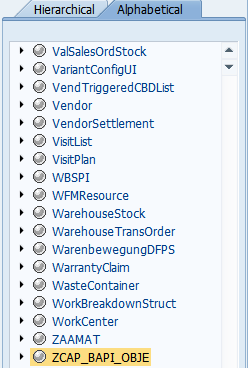
**Step # 36:** will get message “Object type generated successfully” in the status bar.



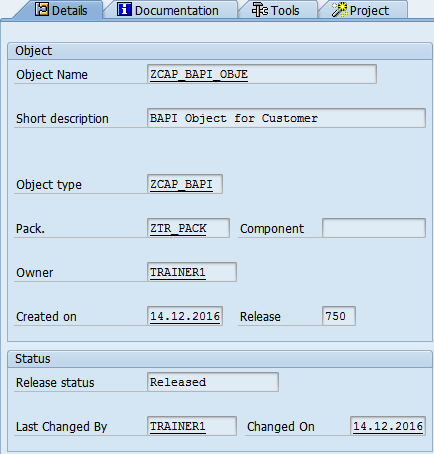
**Step # 37:** Go to BAPI T-code.



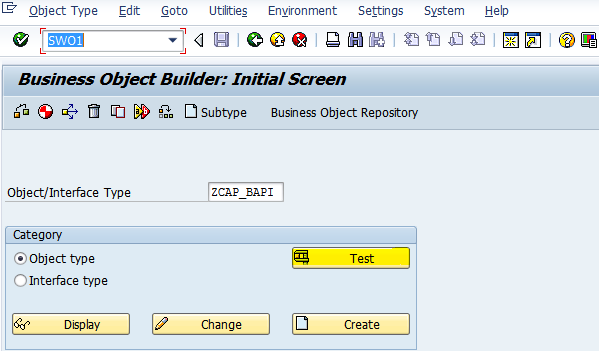
**Step # 38:** Select the Alphabetical tab and search to cross check the customized BAPI found or not?



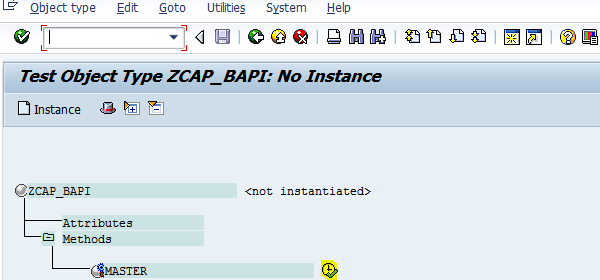
**Step # 39:** Check the BAPI Object related details in the right side window.



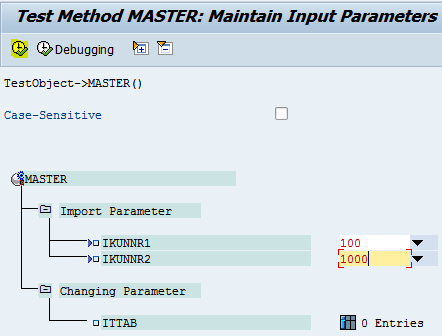
**Step # 40:** Go to SWO1 T-code and provide the object name and click on test.



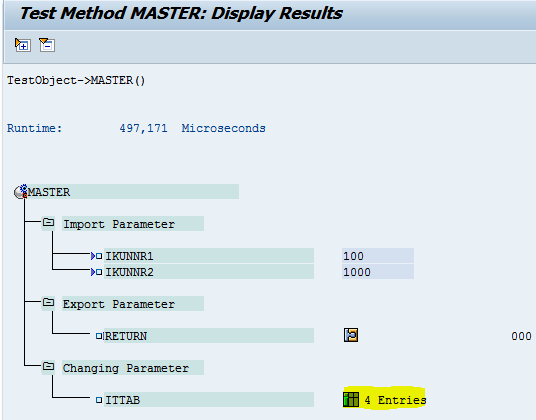
**Step # 41:** Click on execute method button.

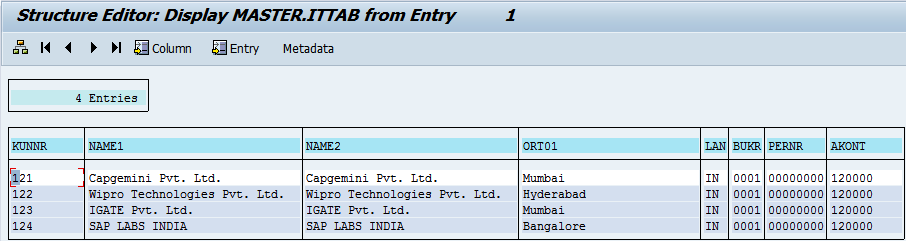


**Step # 42:** Provide the customer input range in the import parameter and click on execute.



**Step # 43:** Click on Entries/Content icon under the changing parameter.



**Step # 44:** Successfully executed. 

You have executed the BAPI. Not SE37 function Module.

Business objects encapsulates BAPI. We have created custom logic to retrieve a few fields.