Wednesday, 08 April 2020 10:49

Implementing DCL with CDS Views and Roles

Written by https://blogs.sap.com/2020/04/09/implementing-dcl-with-cds-views-and-roles/

Hi Readers,

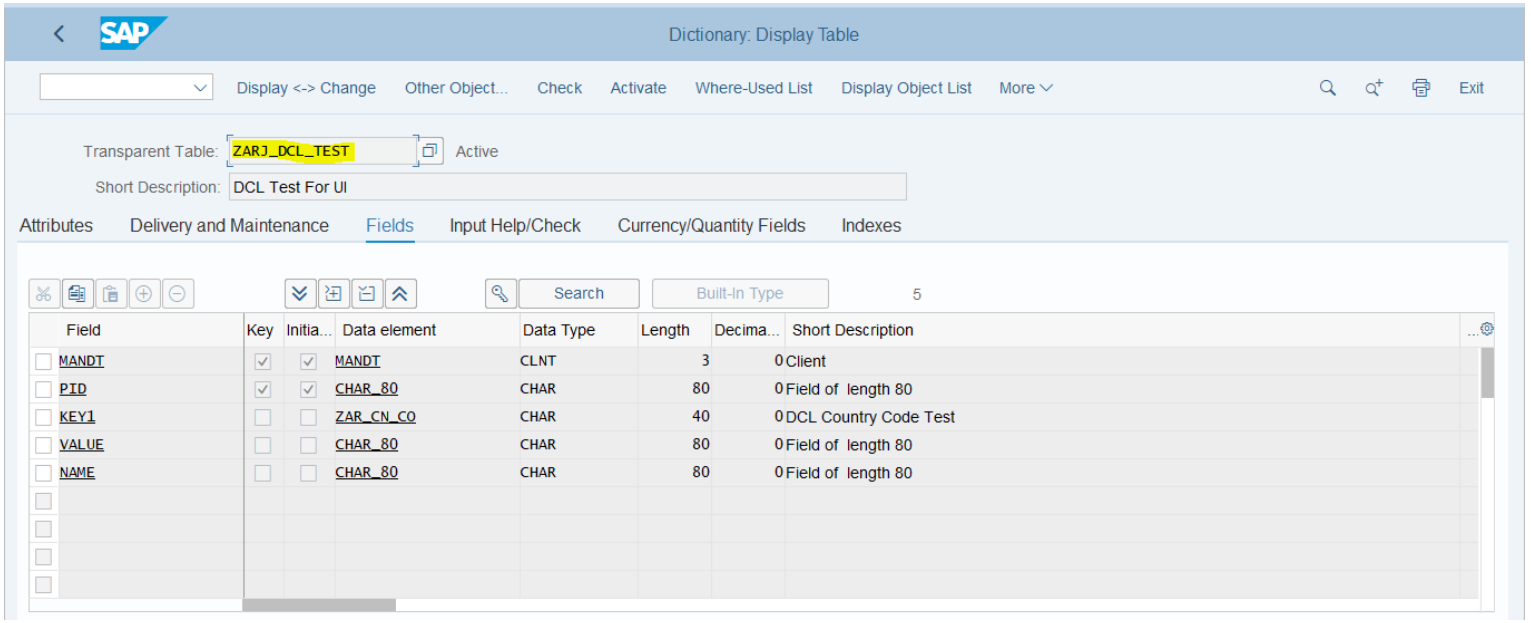
I am going to write here about how to use the concept of DCL with CDS views and using roles to provide row based authorizations in UI.

So,

DCL stands for Data Control Language. It provides an access control mechanism to restrict the results returned by the CDS view from the database according to conditions. With the help of roles (PFCG), we can also use the same DCL and restrict the results for different categories of users.

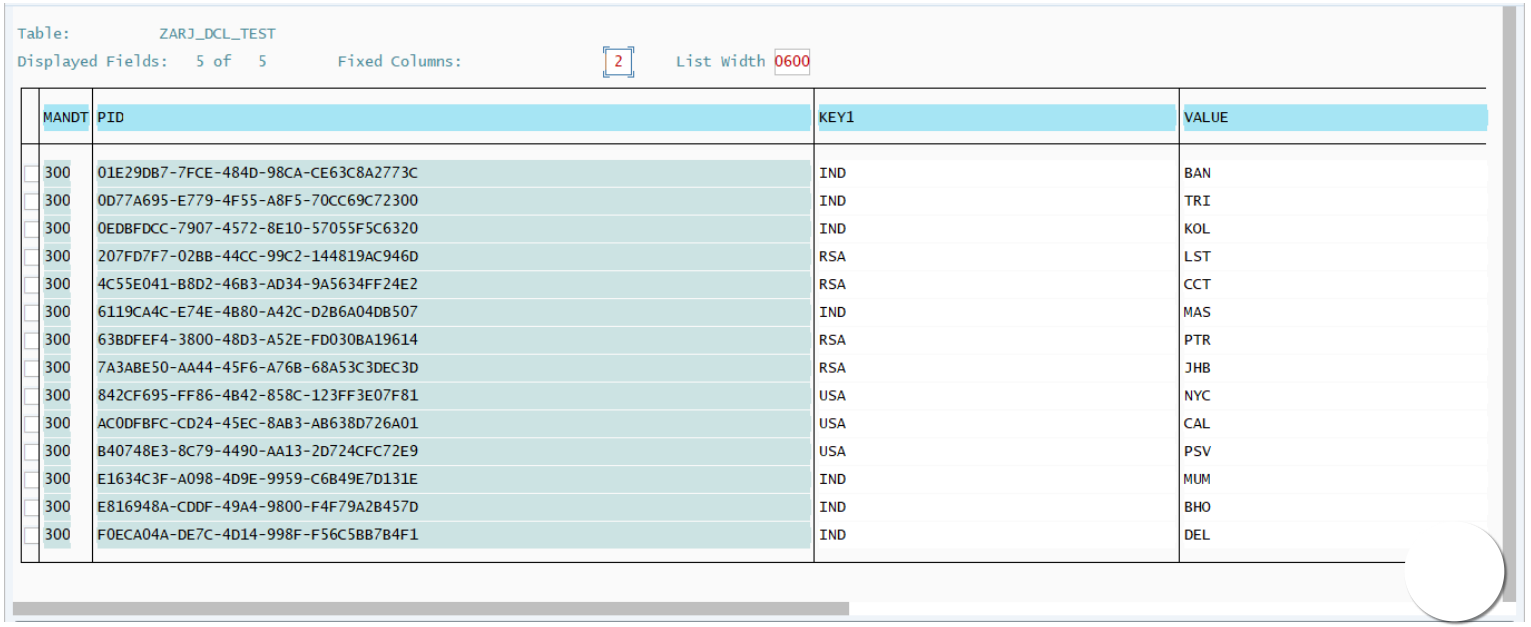
Let us look into the process, with the help of an example.

Suppose I have the below database table:



ZARJ_DCL_TEST – The table with the fields MANDT, PID, KEY1, VALUE and NAME.

Now, lets check the data entered for the above table:



Here in the KEY1 field we can see that we have three country codes, RSA, IND and USA. Now, our aim is to display records with only one or two of these country codes to the user from UI.

So, in the next step, we can create an Implementation CDS View (ZDCL_I_CO) from this table.

```
@AbapCatalog.sqlViewName: 'ZAR_I_DCL' @AbapCatalog.compiler.compareFilter: true @AbapCatalog.preserveKey: t
```

Now, consume this implementation view in our next consumption view (ZDCL_C_CO):

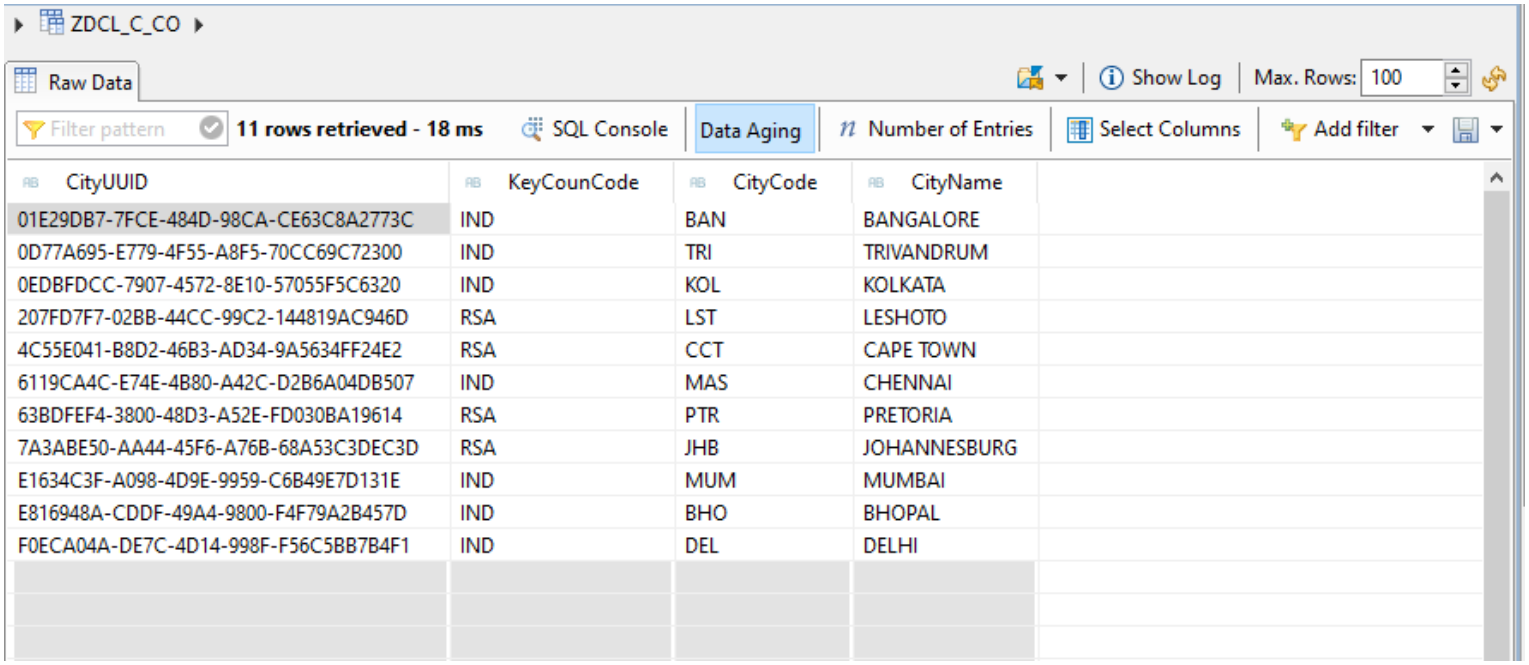
```
@AbapCatalog.sqlViewName: 'ZAR_C_DCL'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Consumption DCL'
@VDM.viewType: #CONSUMPTION
@OData.publish: true
@Search.searchable: true
define view ZDCL_C_CO as select from ZDCL_I_CO {
    key CityUUID as CityUUID,
    @Search.defaultSearchElement: true
    @UI.identification: [{position: 10}]
    @UI.selectionField: {position: 10}
    @UI.lineItem: { position: 10}
    @EndUserText.label: 'Country Code'
    KeyCounCode as KeyCounCode,
    @Consumption.filter: {selectionType: #SINGLE, multipleSelections: false}
    @UI.selectionField: {position: 20}
    @UI.identification: [{position: 20}]
    @UI.lineItem: { position: 20}
    @EndUserText.label: 'City Code'
    CityCode as CityCode,
    @UI.lineItem: { position: 30}
    @EndUserText.label: 'City Name'
    CityName  as CityName
}

```

Here is the code:

```
@AbapCatalog.sqlViewName: 'ZAR_C_DCL' @AbapCatalog.compiler.compareFilter: true @AbapCatalog.preserveKey: t
```

Now, lets check the output by viewing it in the data preview in eclipse:



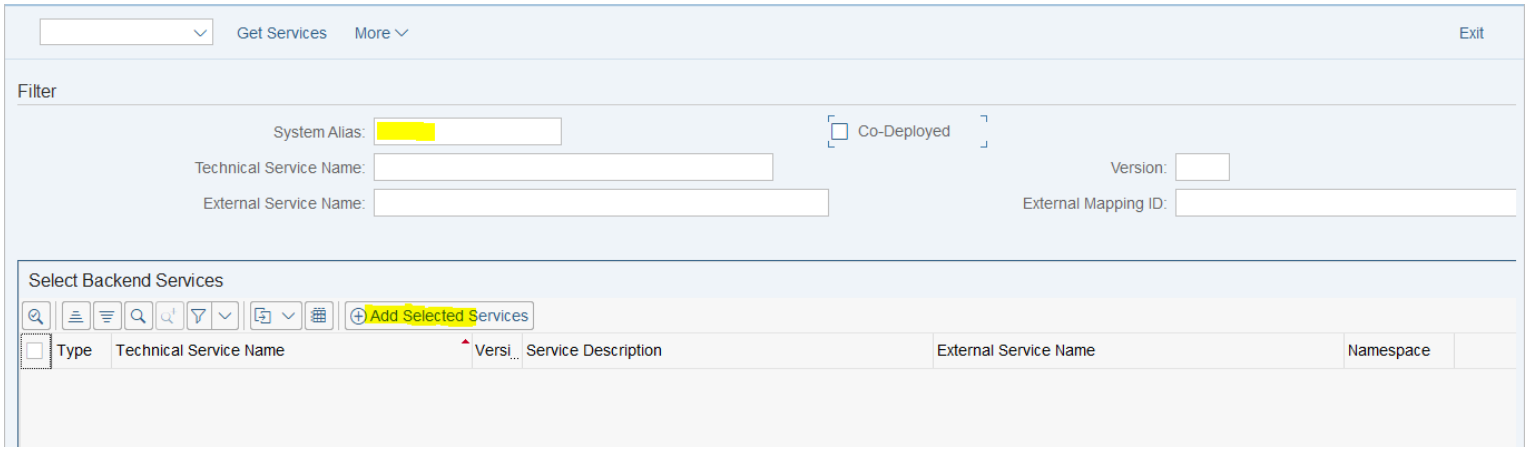
CityUUID	KeyCounCode	CityCode	CityName
01E29DB7-7FCE-484D-98CA-CE63C8A2773C	IND	BAN	BANGALORE
0D77A695-E779-4F55-A8F5-70CC69C72300	IND	TRI	TRIVANDRUM
0EDBFDCC-7907-4572-8E10-57055F5C6320	IND	KOL	KOLKATA
207FD7F7-02BB-44CC-99C2-144819AC946D	RSA	LST	LESHOTO
4C55E041-B8D2-46B3-AD34-9A5634FF24E2	RSA	CCT	CAPE TOWN
6119CA4C-E74E-4B80-A42C-D2B6A04DB507	IND	MAS	CHENNAI
63BDFEF4-3800-48D3-A52E-FD030BA19614	RSA	PTR	PRETORIA
7A3ABE50-AA44-45F6-A76B-68A53C3DEC3D	RSA	JHB	JOHANNESBURG
E1634C3F-A098-4D9E-9959-C6B49E7D131E	IND	MUM	MUMBAI
E816948A-CDDF-49A4-9800-F4F79A2B457D	IND	BHO	BHOPAL
F0ECA04A-DE7C-4D14-998F-F56C5BB7B4F1	IND	DEL	DELHI

Now from here we need to follow the below steps to create a Access Control (DCL)

- Create a Authorization Field and Authorization Object
- Create a Access Control in Eclipse for the desired CDS view
- Create a PFCG role and assign it to the user with authorizations
- Create a Fiori App to preview the output (Optional as the output can be viewed in the CDS level itself)

Creating an OData Service and a Fiori Worklist App

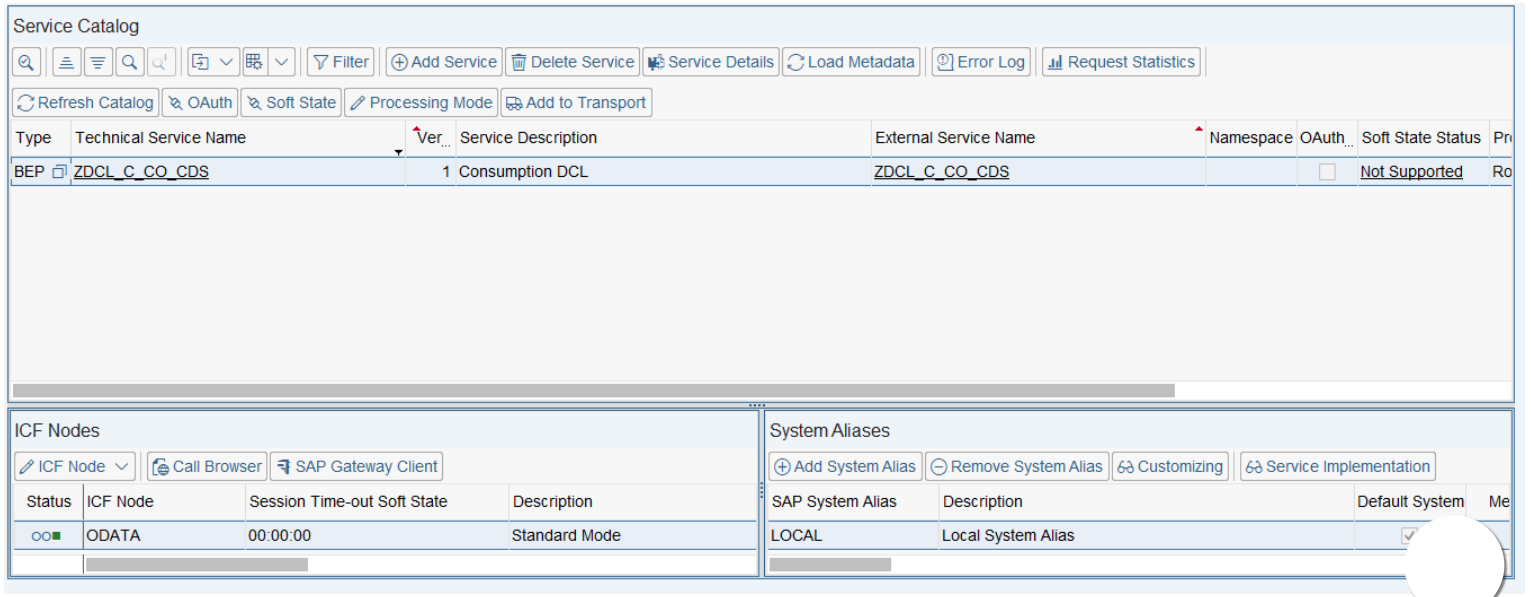
As I have mentioned before this step is optional.We can activate the OData Service generated from the above consumption view (due to the @OData.publish: true annotation in line 7). To do this, go to the TCode /n/iwfnd/maint_service and click on the add service button. The below screen will come up.



Here enter the system alias as “LOCAL” and enter the CDS view name with both preceding and succeeding stars in the Technical Service Name and click on get services. Our CDS generated service should come in the Select Backend Services table below.

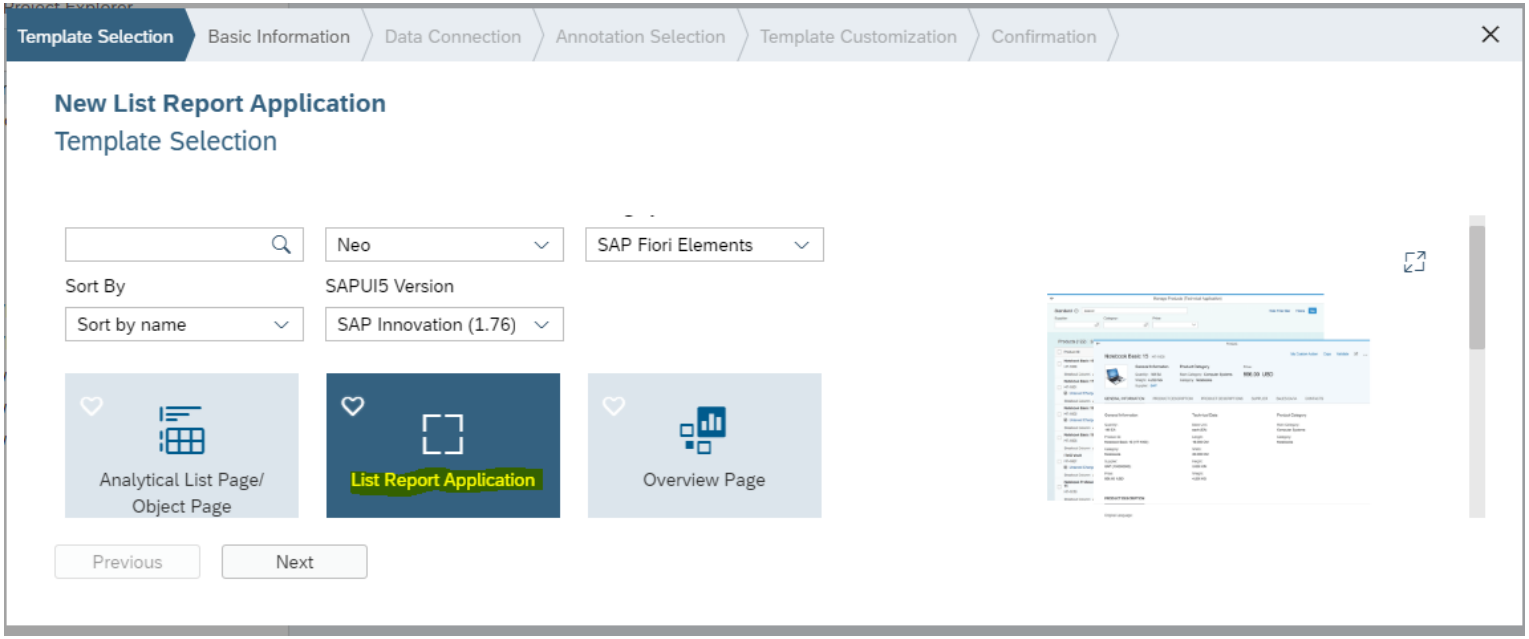
Select the checkbox beside the service in the table and click on the add selected services button to add the service. It would prompt a dialog for selecting the transport request for the MPC and DPC classes. Select an transport request or click local object and save it.

Now go to the previous screen and select the filter and enter the service name added on the previous screen and click enter. In this case, it should display the service as below:

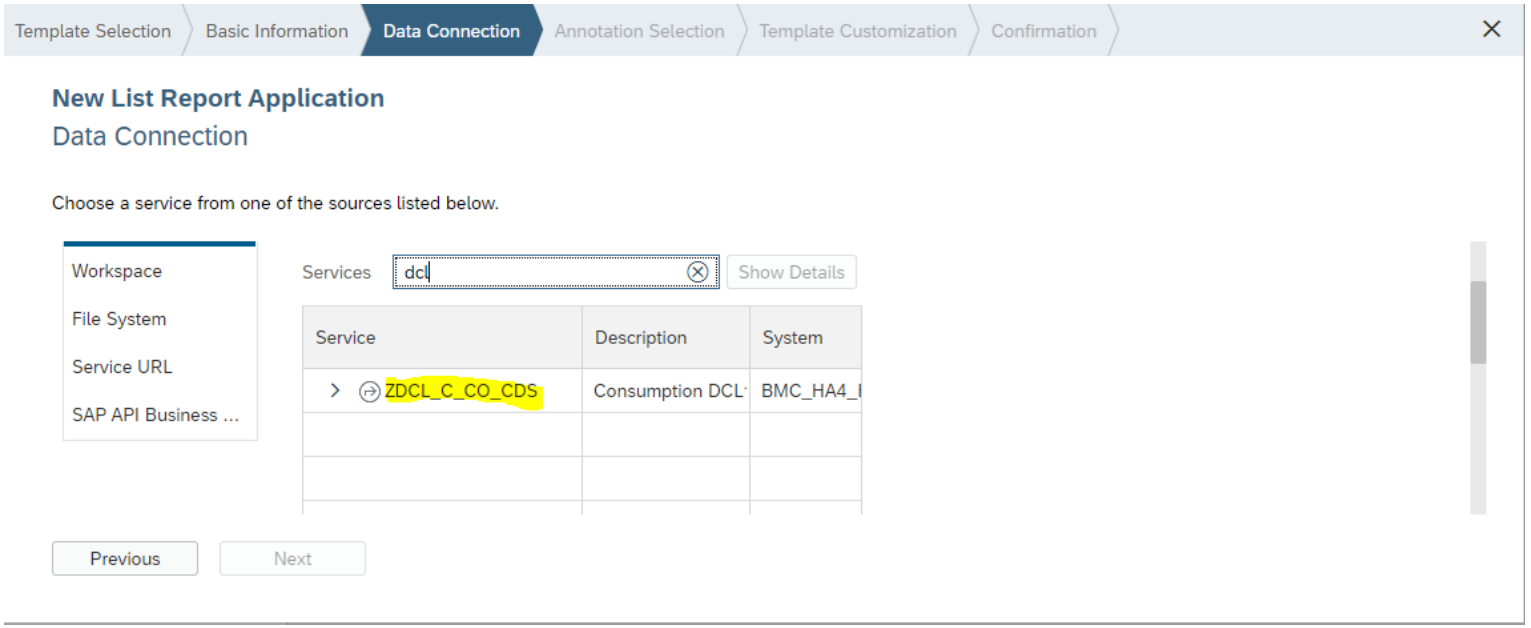


Now in our WebIDE, we can create a List report application with this OData service, to look at the data that is being fetched:

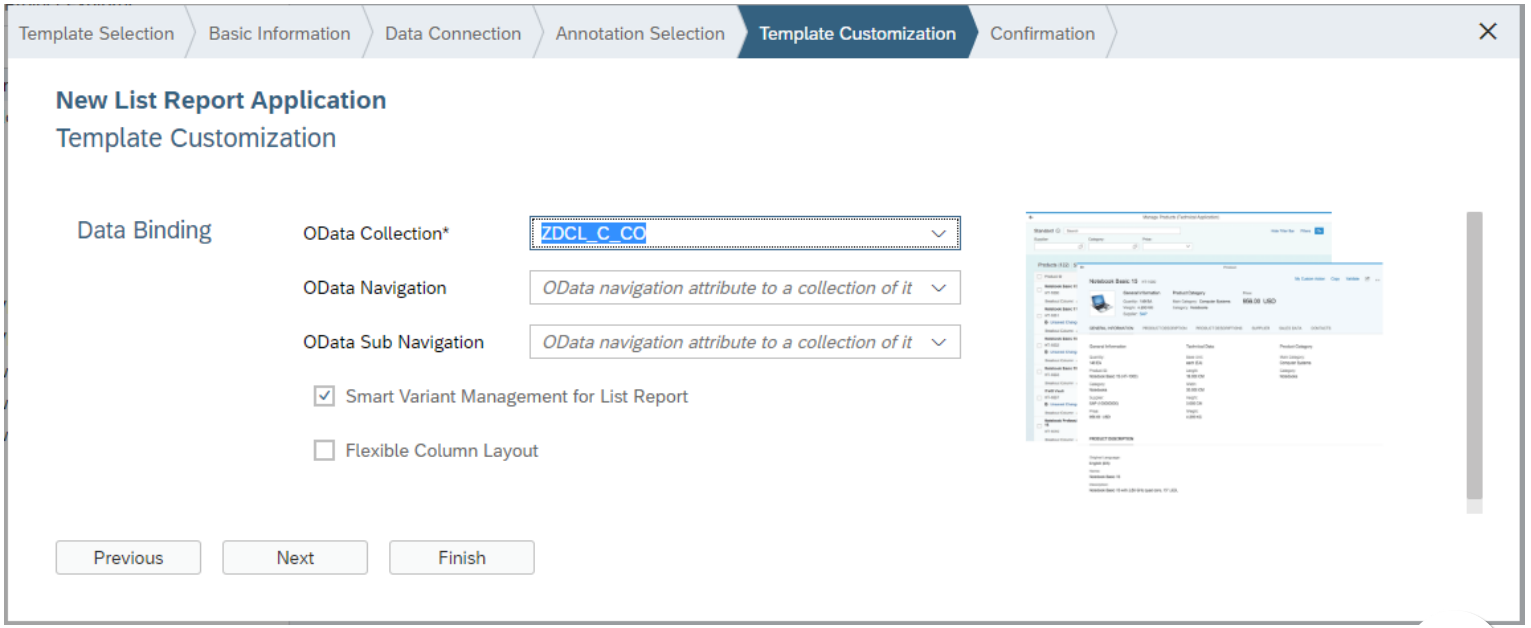
Select the List Report Application in the list of templates-



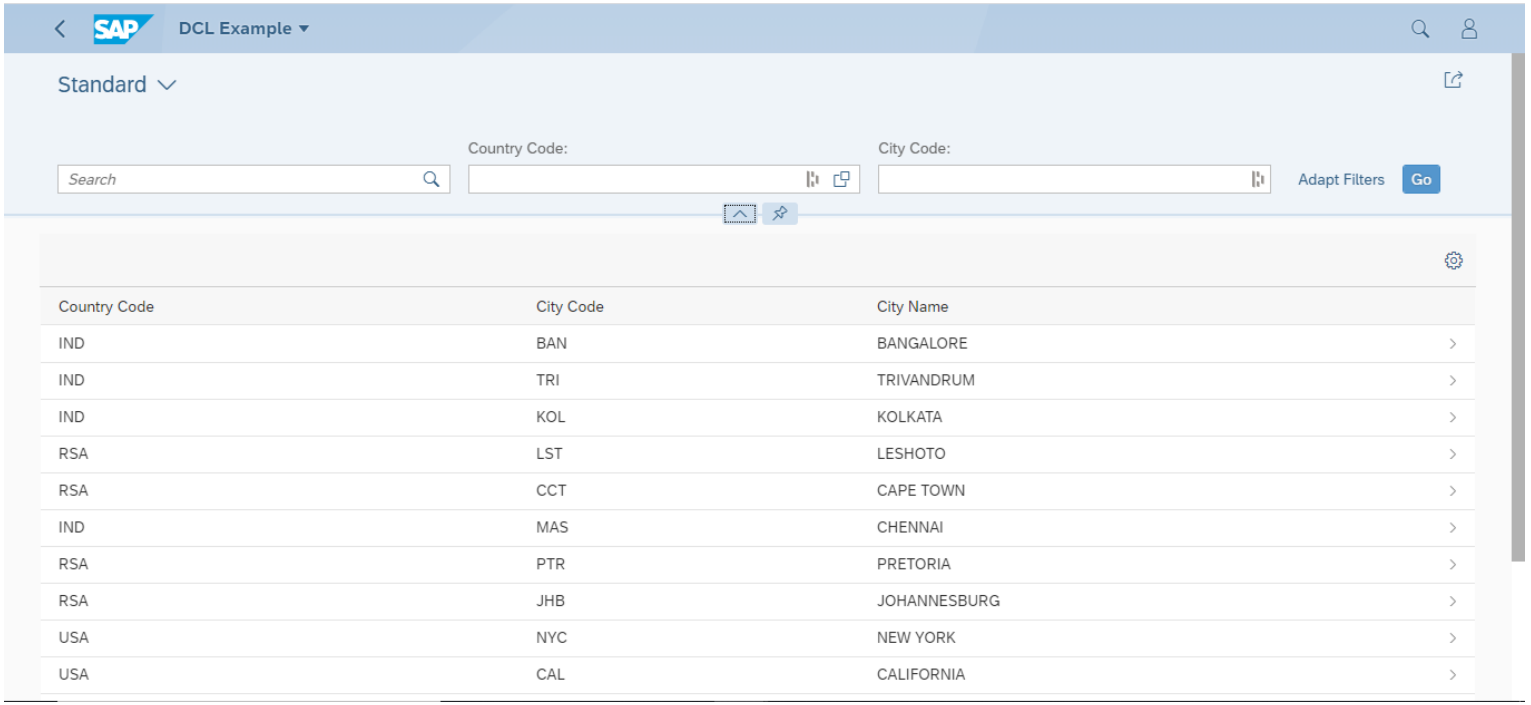
Select the proper OData service & in the next screen select all the annotations-



Select the header collection entity set (This is responsible for the data that is being populated in the first view table)-



Now, click on finish and run the application with proper run configurations (preferably run the application from the component instead of the index.html). The below screen should appear with both the smart filters and the smart table.



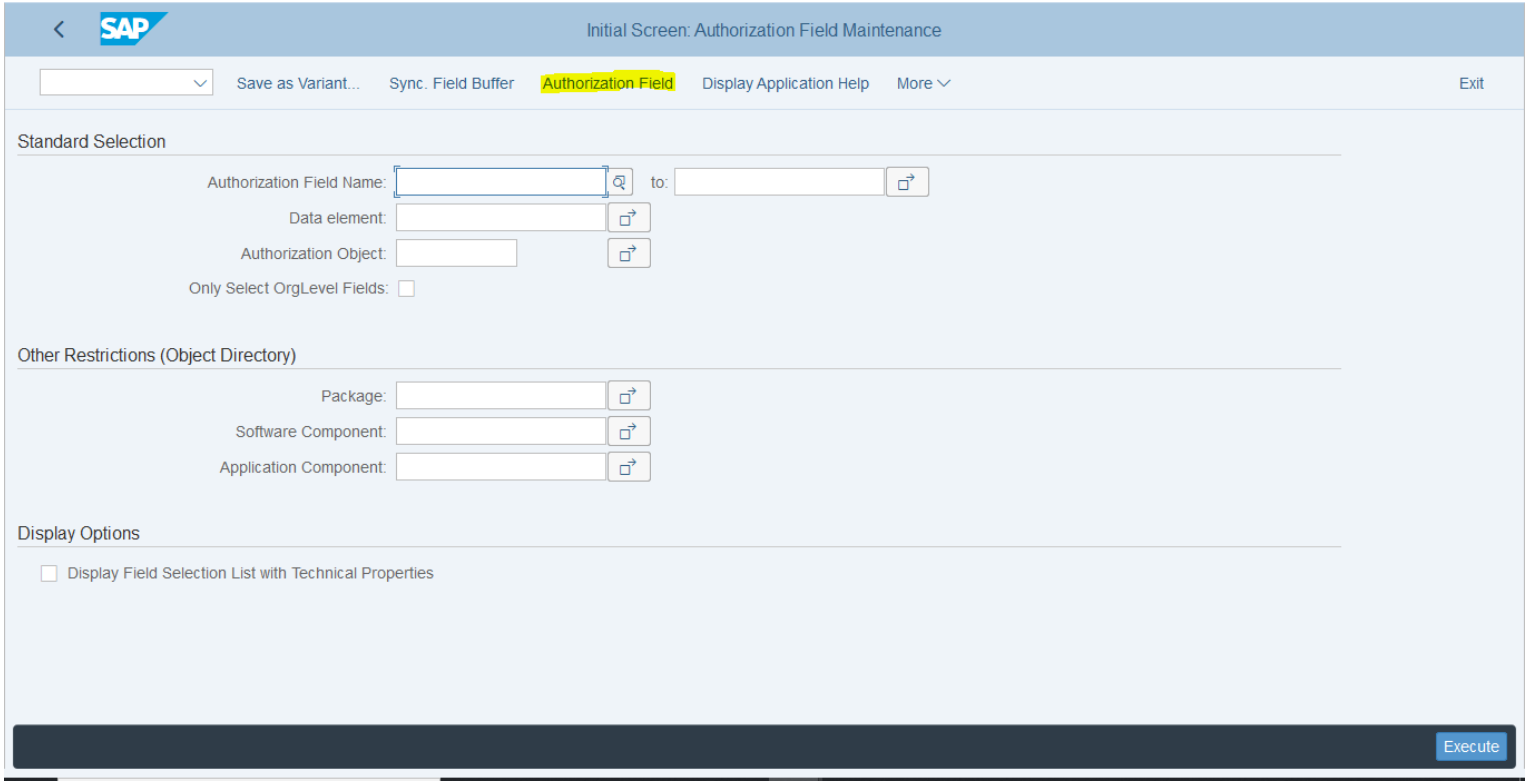
We can see that all the country codes and all the data is visible in the UI.

Creating a Authorization Object and Authorization Field

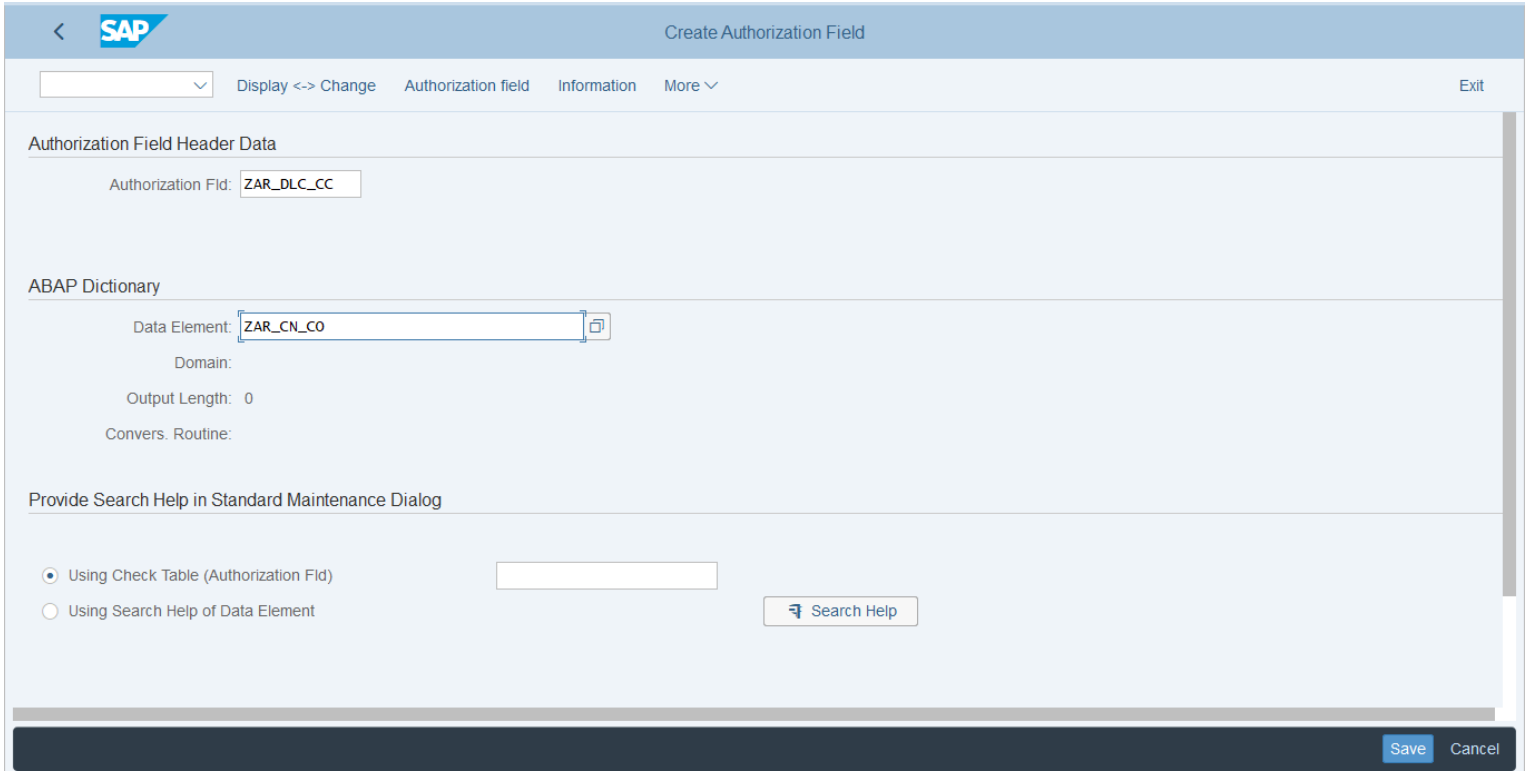
To do this, we make use of the TCodes SU21 and SU20 respectively.

Authorization Field: The *authorization fields* contain data element fields that are in the form of single values or range value and this value sets which are used for authorizations.

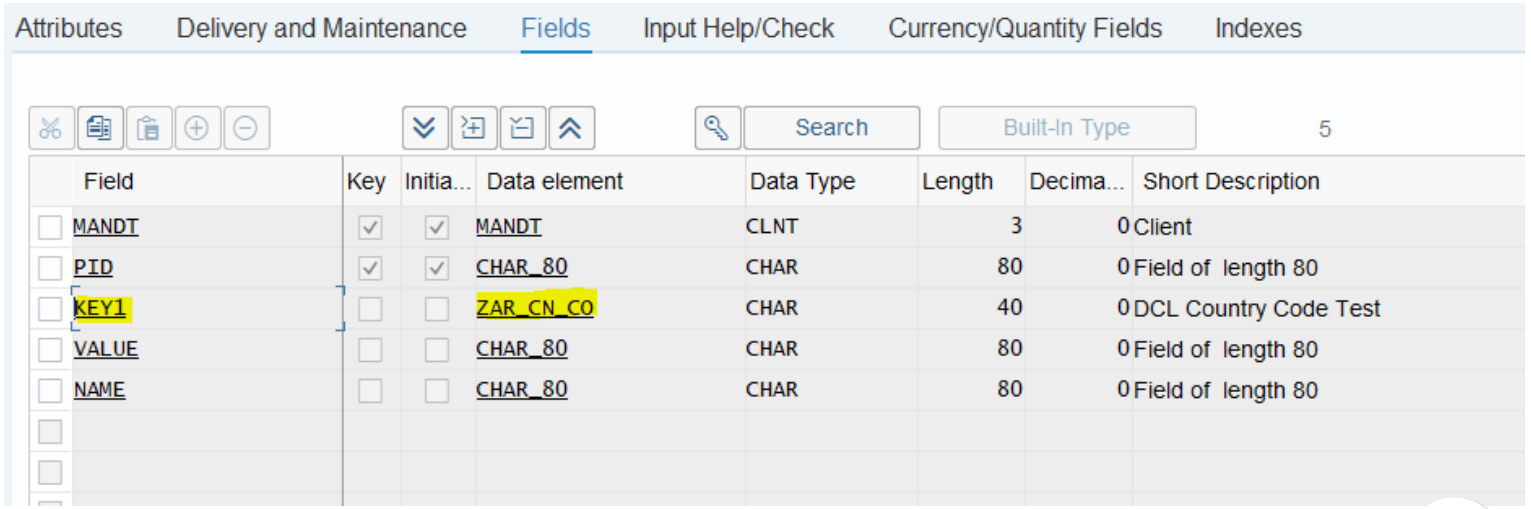
To create an authorization field, we go to the TCode SU20. Click on the Authorization Field button on the top menu bar.



In the next screen, enter the desired authorization field name (ZAR_DLC_CC), and the name of the data element you wish to provide authorization for. In our case the data element is ZAR_CN_CO, for the field KEY1 in our se11 table.



Screenshot of our se11 table field for which we wish to add authorization to:



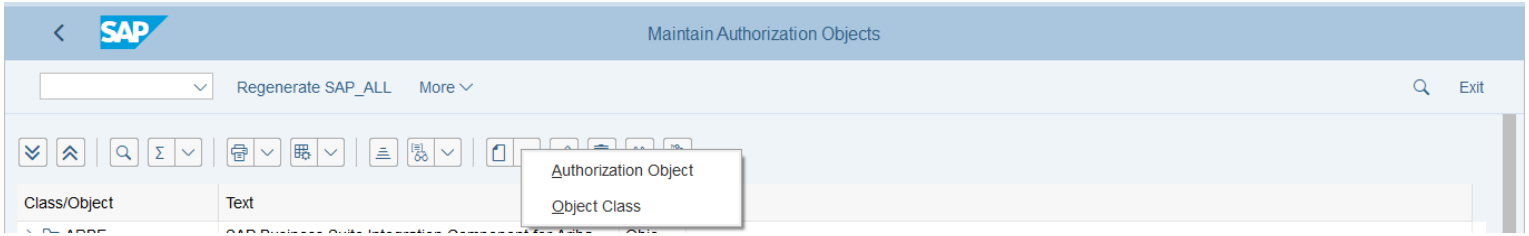
Authorization Class/Object: The authorization class is an group of one or more authorization obje
Authorization object is a place where the configurations regarding authorizations and permissions are

set up and initialized against the authorization fields.

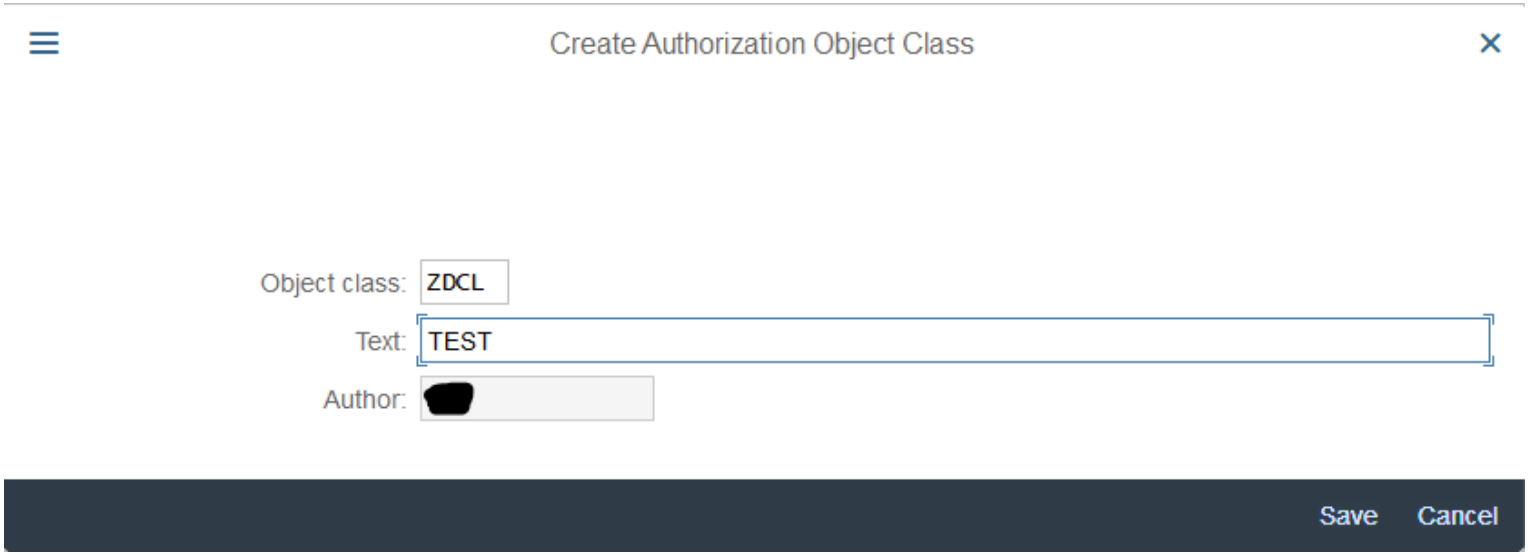
We do not need to create the authorization class every time we create the authorization object. It can be done only once.

To create the Authorization Class and object, we go to the tcode SU21. All our DCL based authorizations can be collectively placed in an authorization class. We can name it as ZDCL.

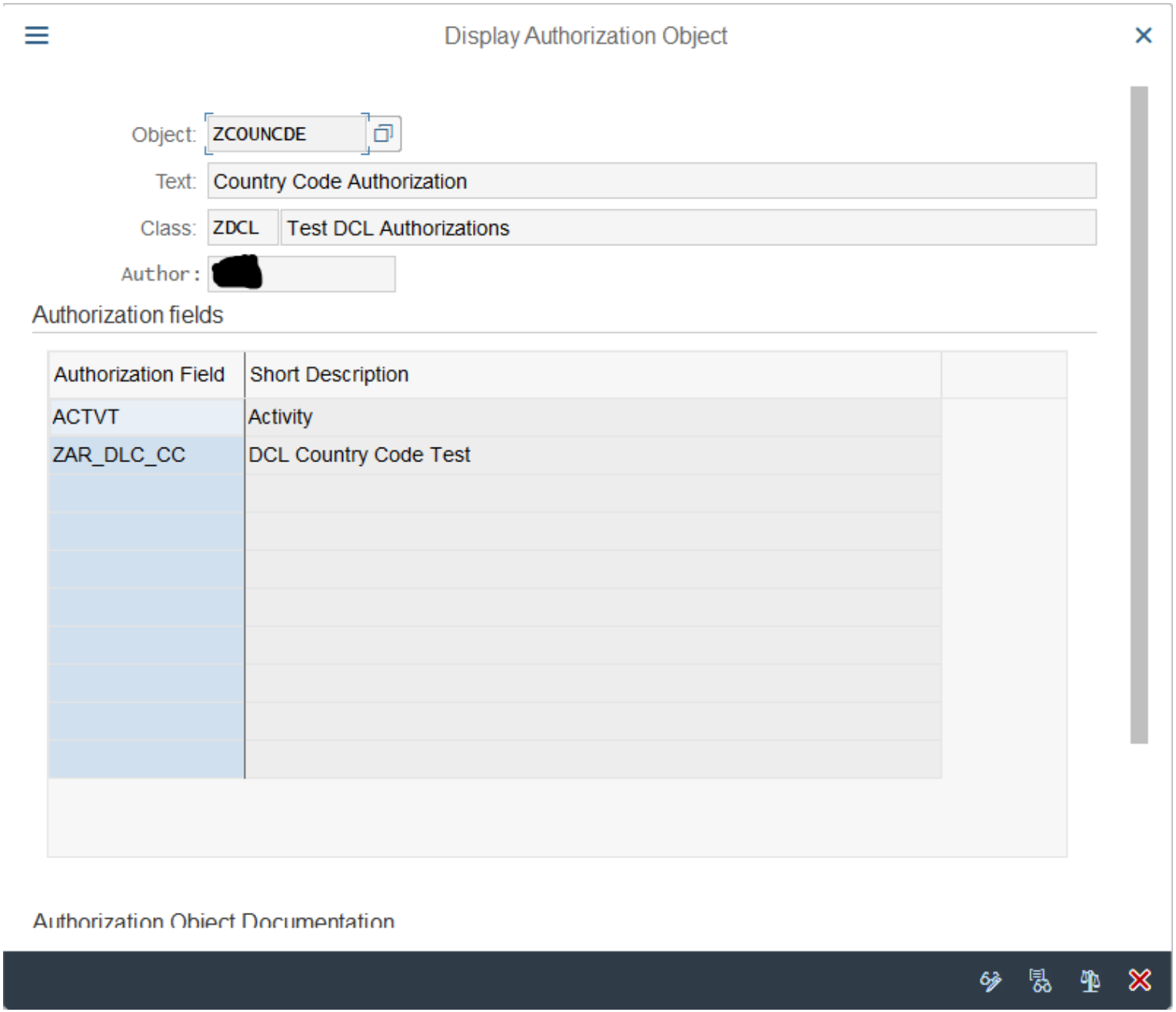
So, click on the new button (paper icon) and click on Object Class.



Enter the details in the below screen and create a class.



Now,under the class, create an authorization object (ZCOUNCDE). Click on the authorization object button in the above screen. Enter the authorization fields and click on save.

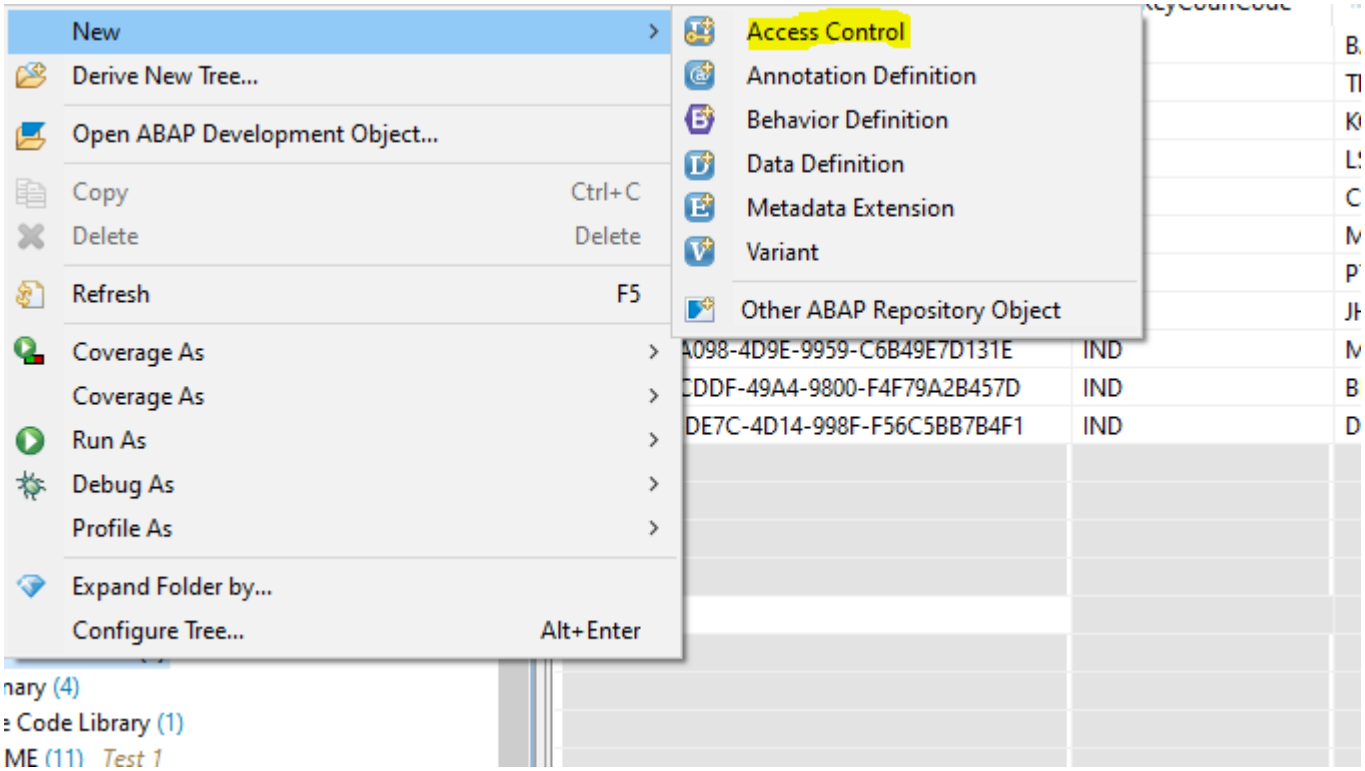


On clicking save, the below screen should appear:

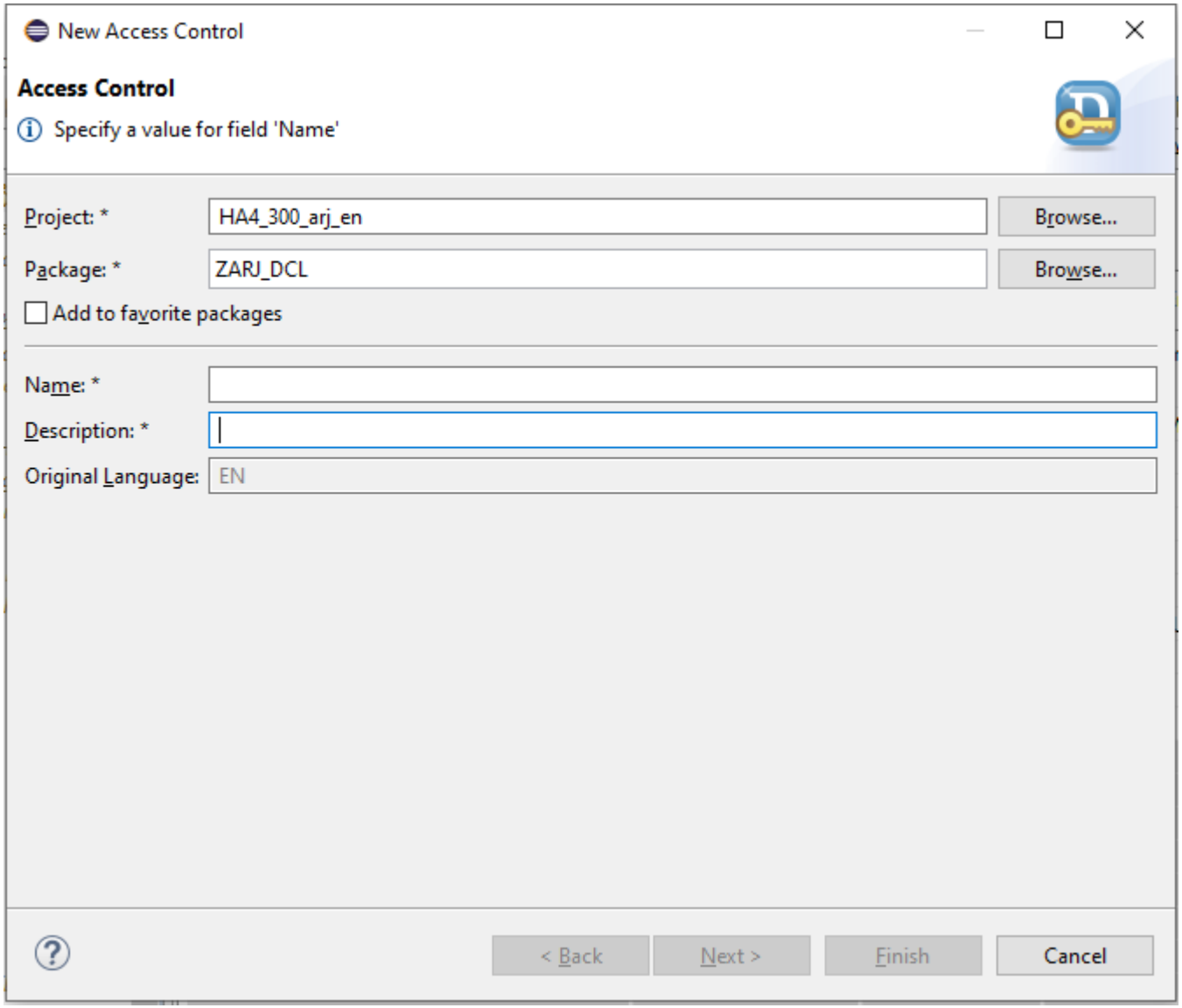
ZDCL	Test DCL Authorizations	Object Class
ZCOUNCDE	Country Code Authorization	Authorization Object

Creating an Access Control with CDS

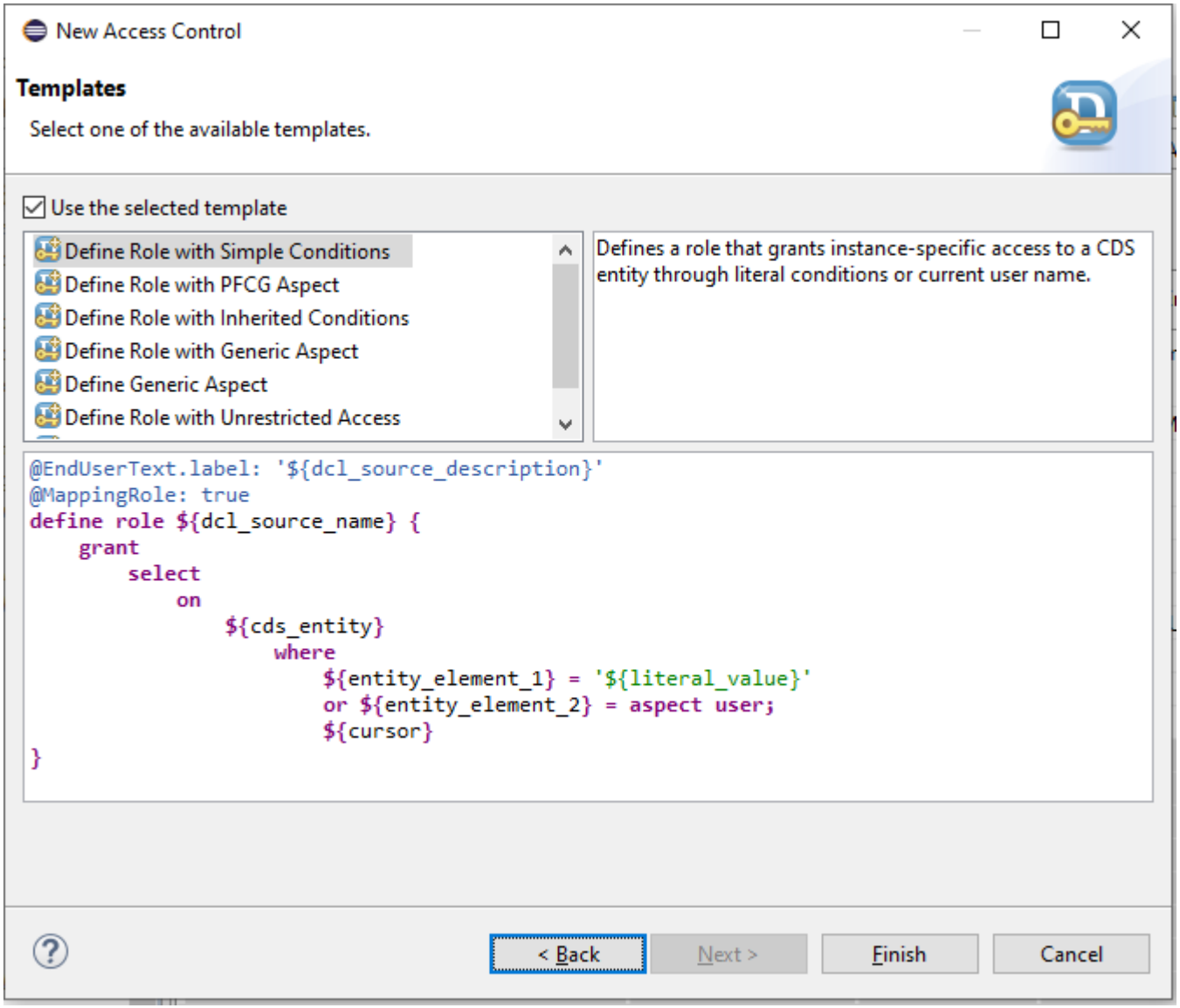
Now that we have an Authorization Object and a Authorization Field, we need to connect them with our CDS view to make it recognize the DCL. To do this, go to eclipse, right click on the package, click new and select Access Control



In the next dialog, Enter the name (ZDCL_C_CO) and description, click on next, in the next screen select an transport request, then click on next.



Now, in the below screen, for this example, we shall use an “Define Role with Simple Conditions” template. Select this and click on finish.



Now add the below code in the newly generated access control:

```
@EndUserText.label: 'Access Control for ZDCL_C_CO CDS View' @MappingRole: true define role ZDCL_C_CO { grar
```

Here the authorization object and authorization field should be same as the one which was defined in the TCodes SU20 and SU21.

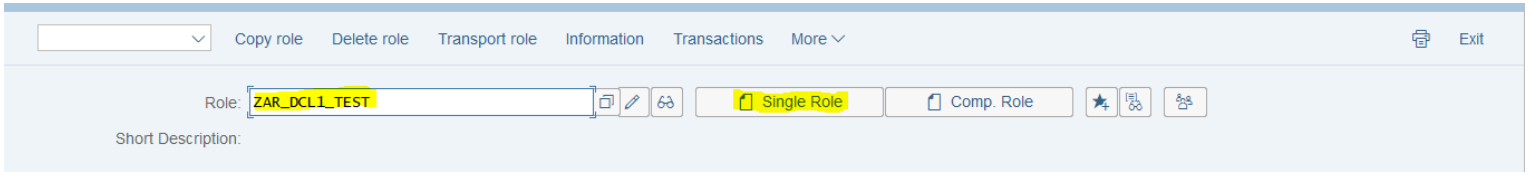
```
@EndUserText.label: 'Access Control for ZDCL_C_CO CDS View'
@MappingRole: true
define role ZDCL_C_CO {
    grant
        select
            on
                ZDCL_C_CO
                    where ( KeyCounCode ) =   aspect pfcg_auth( ZCOUNCDE, ZAR_DLC_CC );

}
```

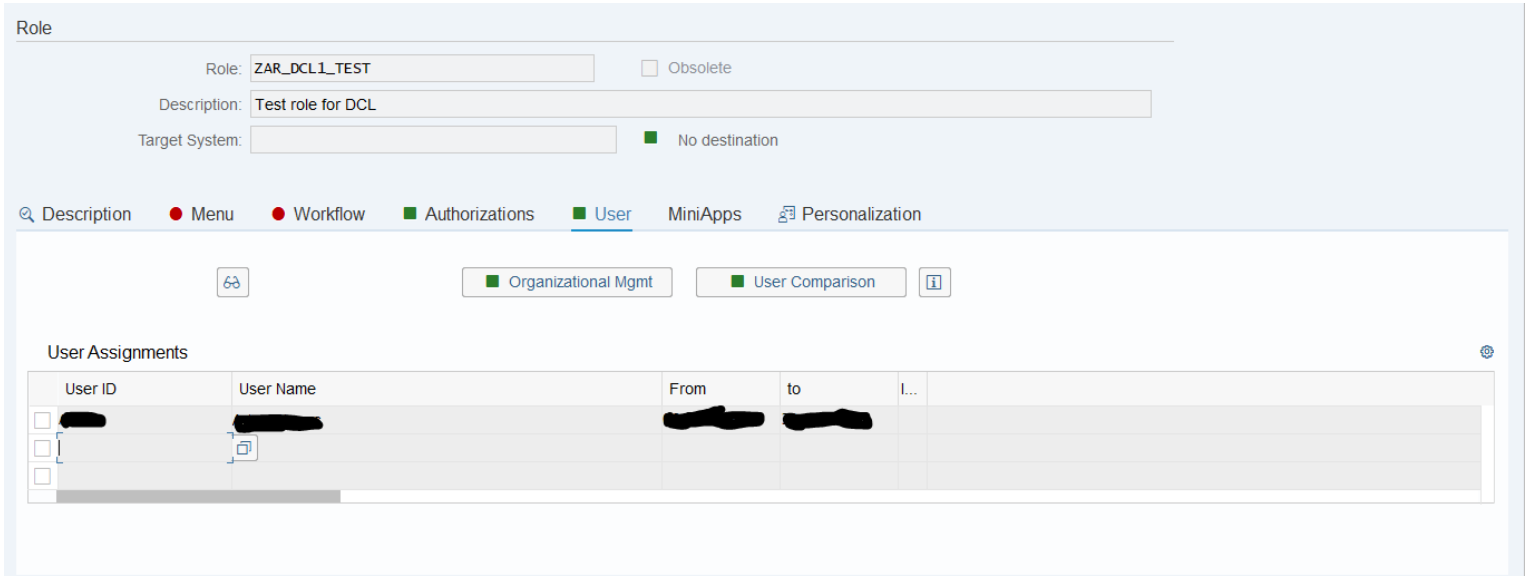
Now save and activate the Access Control.

Creating a role with the authorization object and assigning it to the user

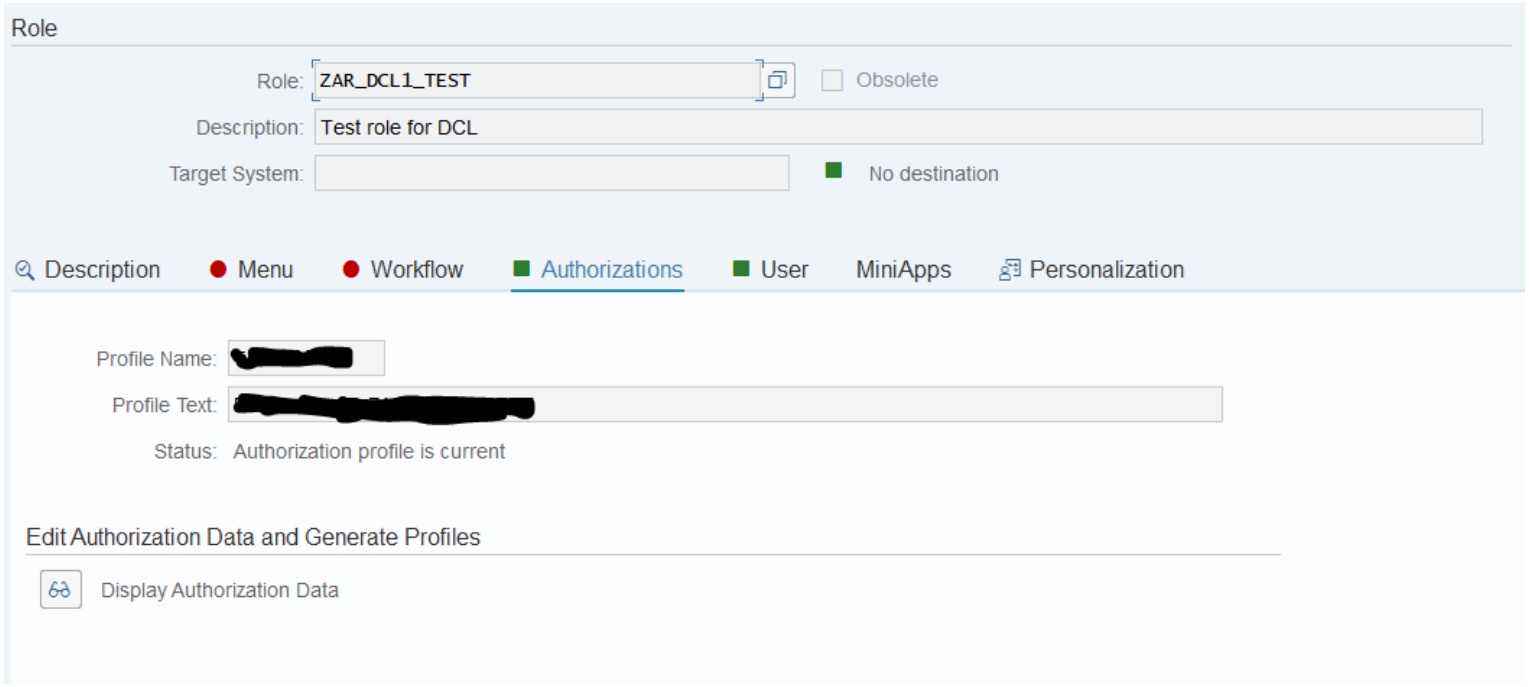
To create a role, we use the TCode PFCG. Open PFCG and enter the desired role name and click on single role-



In the next screen enter the short description and go to the users tab. Click on edit and assign the user ids which are going to test and use this application.

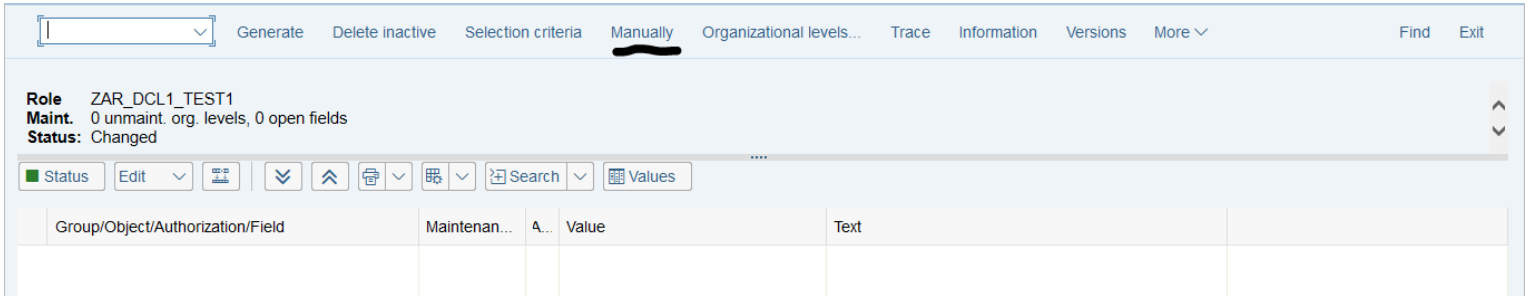


Now go to the authorizations tab and click on display authorization data:

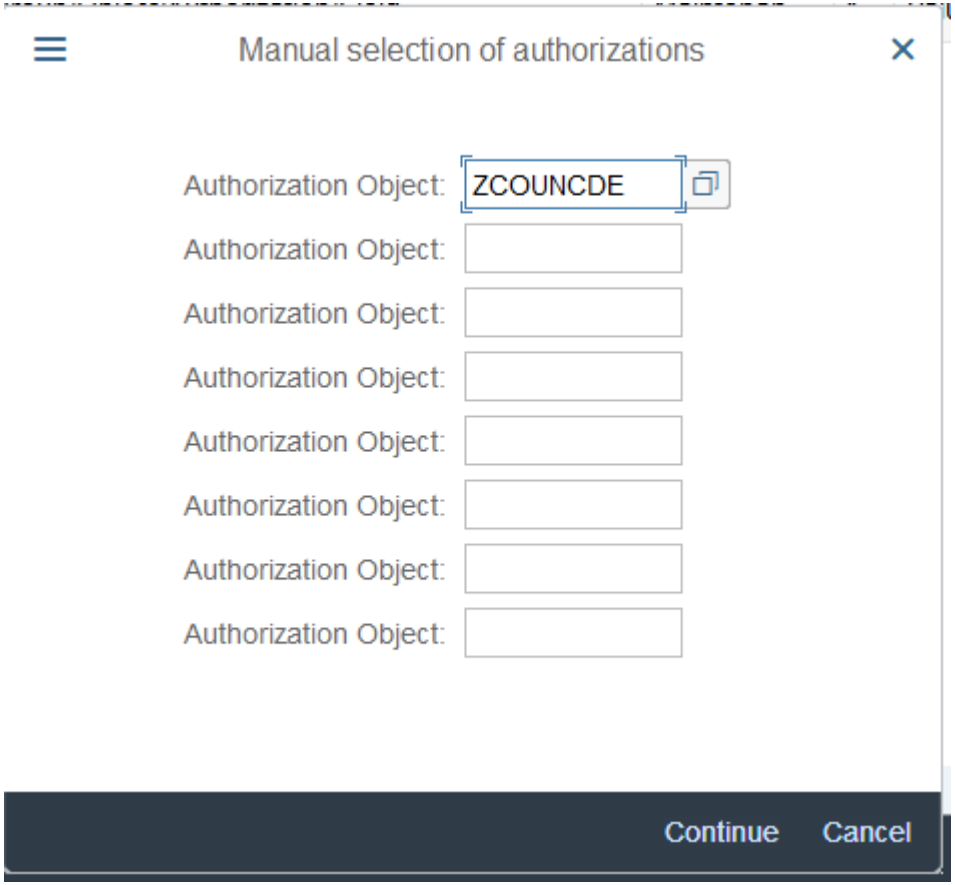


In the next screen, we are going to add our authorization object and assign what values we requir
displayed for the users having this role.

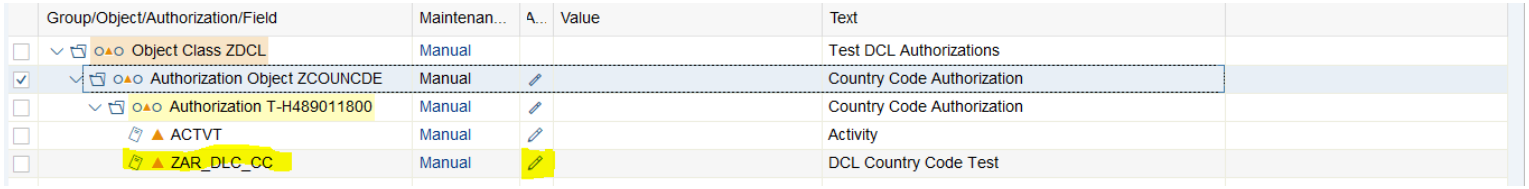
So, in the next screen, if there is any popup requesting for any template click on “Do not select any template”, the in the menu bar, click on the manually icon. This helps us in manually adding the authorization objects of our choice.



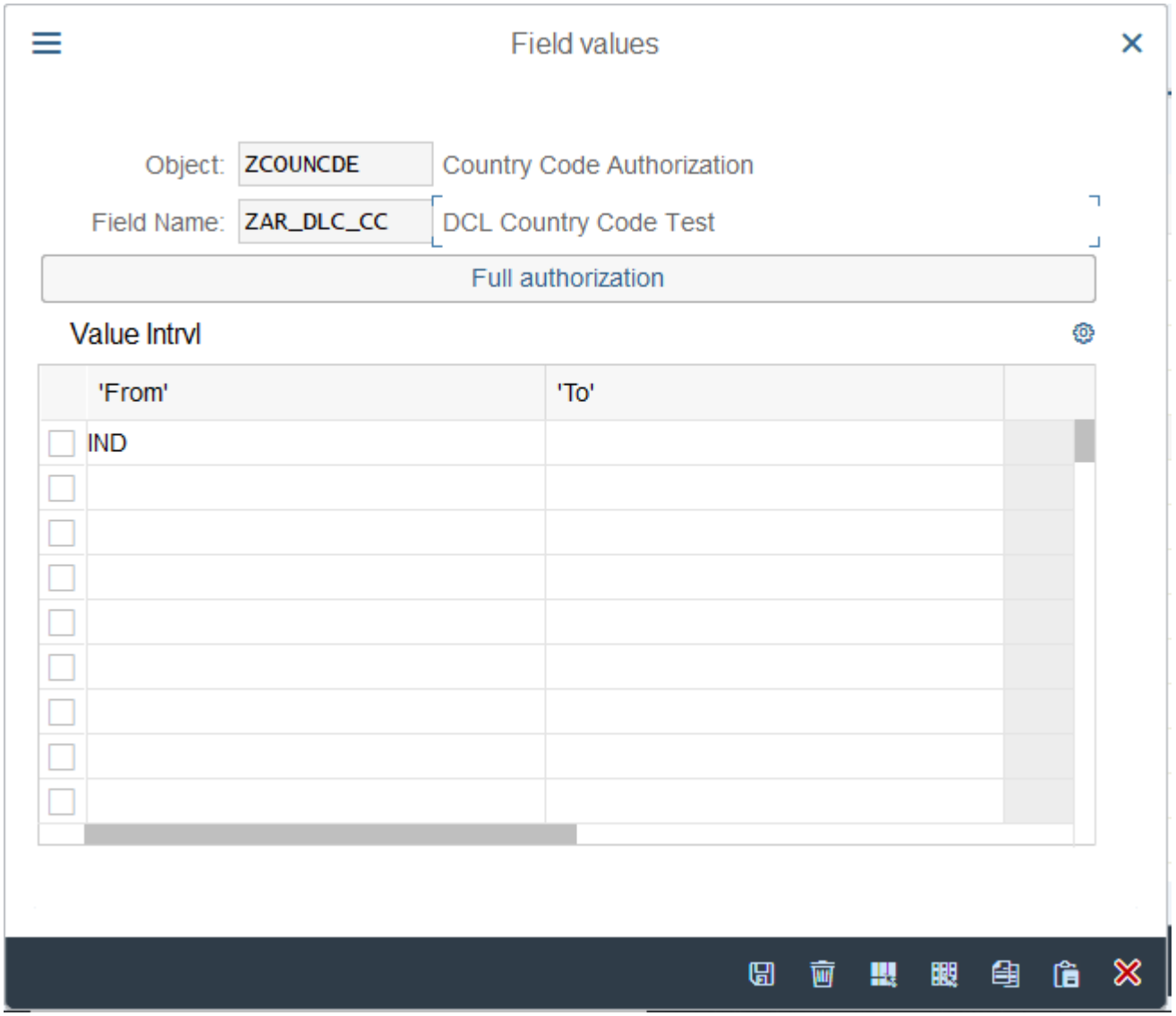
It will open a popup. There add the authorization object that we have created before and click on continue.



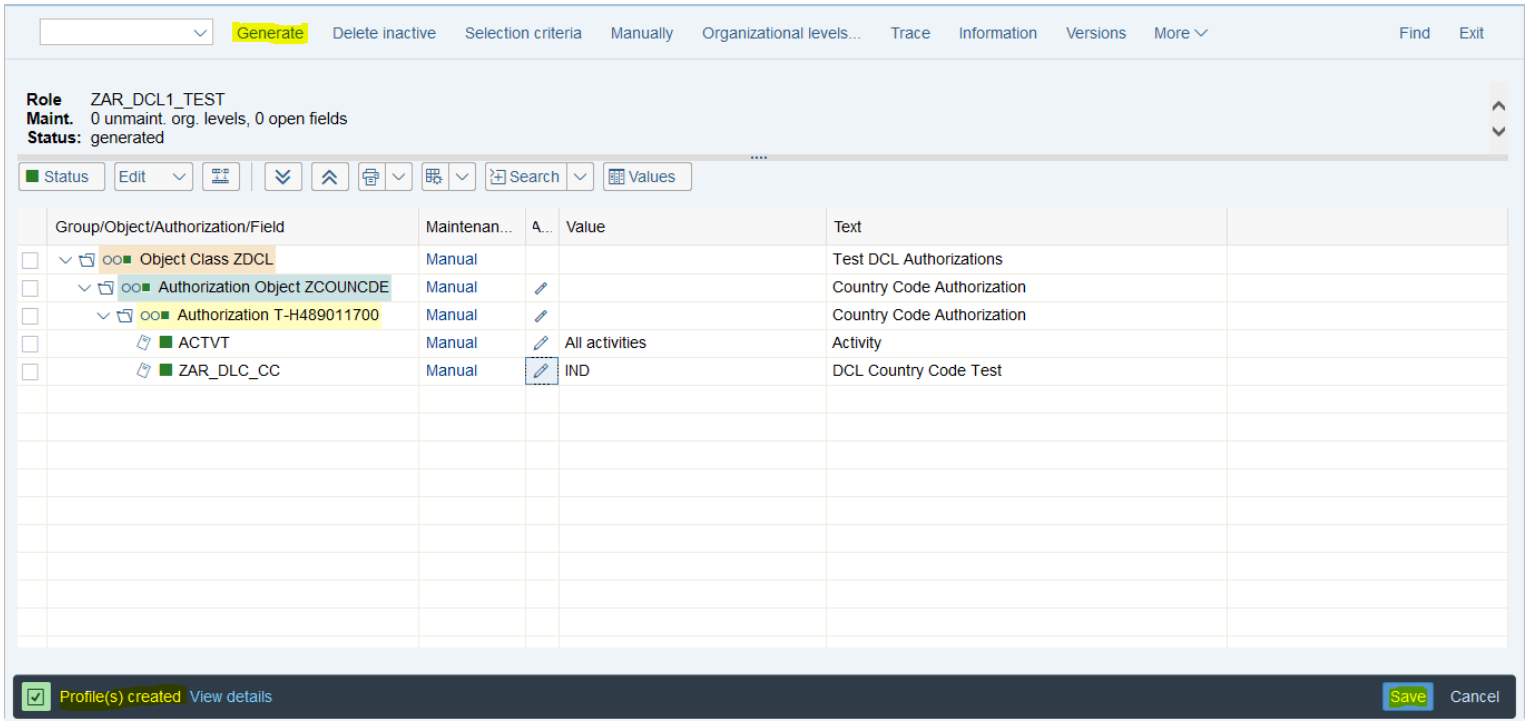
Now, expand the sub tree, and edit the desired authorization field to add restrictions.



To add full authorization enter * or click on full authorization. Else, we can manually enter values or ranges that we require to only appear for the particular user.

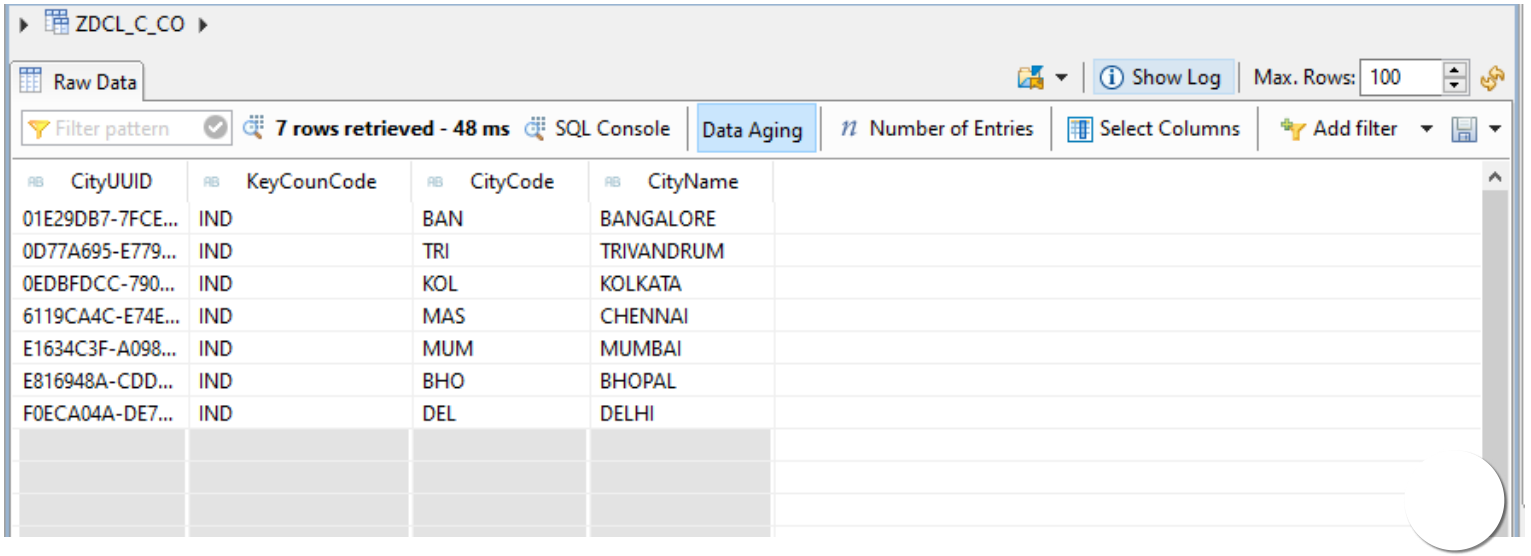


In our case, we need to display the records with country code “IND”. So, we can enter the same in the from column and click on save. Now, save the authorization data and click on generate.

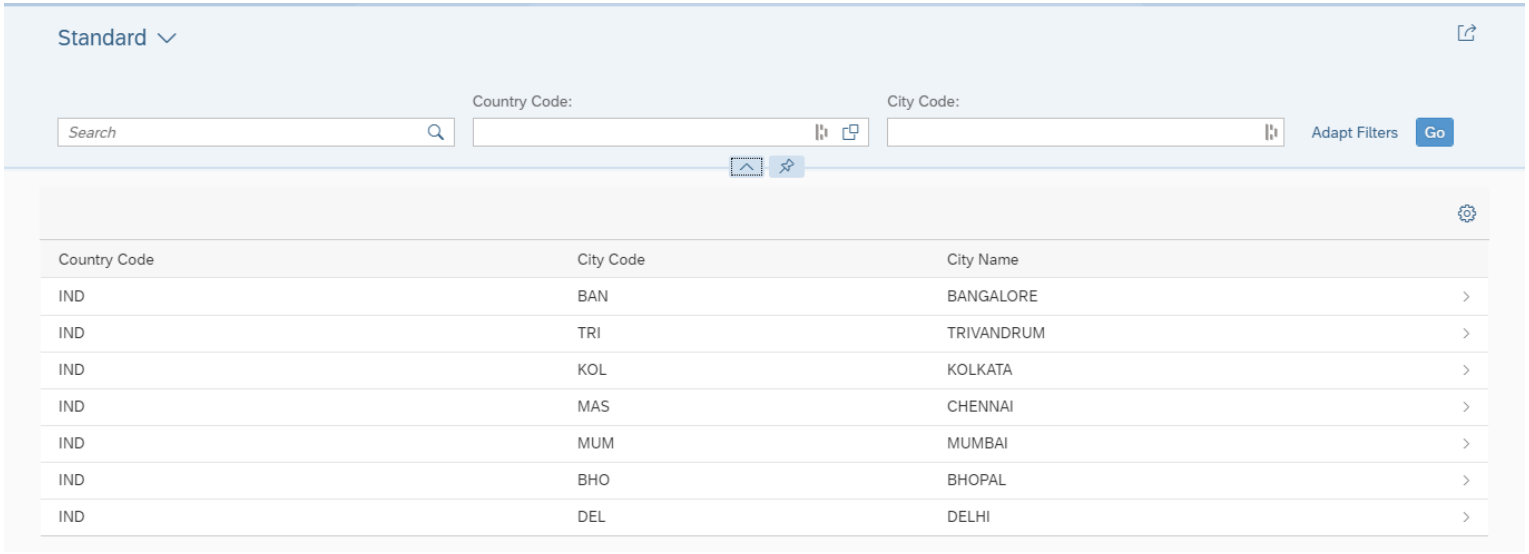


Now, once the profile is generated successfully, if we have assigned the role for our user, we can test the DCL using both the CDS and the front end UI application.

In case of CDS, run the consumption view using the data preview:



Also, now we can go to the browser and refresh the FIORI application:



We can see that in both the cases, we only get the records having country code as IND. Thus our DCL is working and adding the authorizations.

Note:

Through this method, we can add authorizations for the display functionality only. To add authorizations for other functionalities such as create, update and delete, we can place our authorization logic in the BOPF determination.

Please feel free to ask any questions in the comments.

Regards.

Published in SAP HANA

More in this category: « SAP HANA Interview Questions And Answers Part 1: How to Create Dynamic Images in SAP Analytics Cloud Story »

Leave a comment

Make sure you enter all the required information, indicated by an asterisk (*). HTML code is not allowed.

Message *

enter your message here...

Name *

enter your name...

Email *

enter your e-mail address...

Website URL

enter your site URL...

I'm not a robot

reCAPTCHA
Privacy - Terms

Submit comment

back to top

'; jQuery('body').append(panel); setTimeout(showPanel, 1200); function showPanel() {
jQuery('#easycookieinfo').slideDown('slow'); } jQuery('.accept a').on('click', function() { var exp_date = new Date();

Accepted
exp_date.setTime(exp_date.getTime() + (1000*3600*24*365)); document.cookie =
'cookie_32af4b2b4053a2018154d37f74abb95f=accepted;expires='+exp_date.toUTCString()+';path=/';
jQuery('#easycookieinfo').slideUp('slow'); return false; }); });

Useful Links

About Us
Terms & Conditions
Disclaimer

Support

Contact Us

Follow us on