SAPCODES

Fiori Element- CDS View with BOPF CRUD

Posted by Siva Prasad

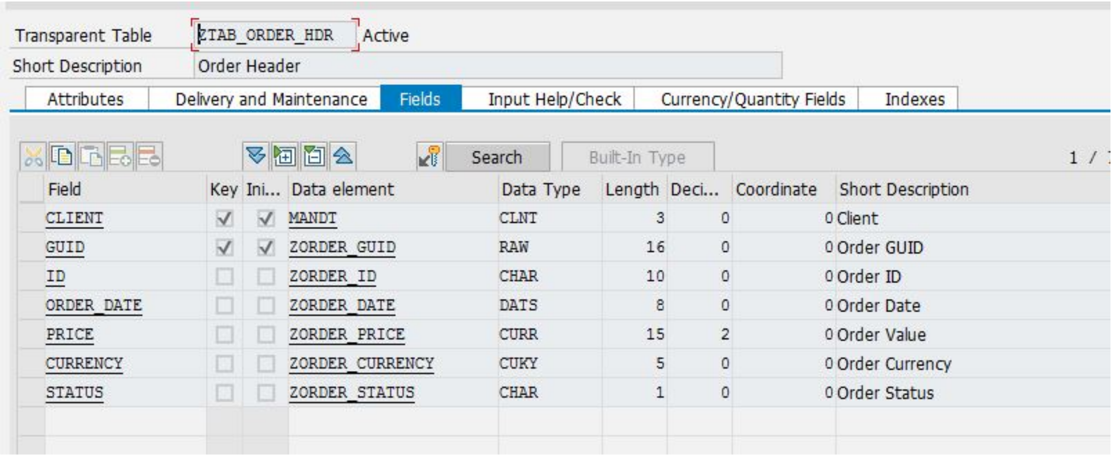
Fiori App- Fiori Element with CDS View with BOPF basic CRUD operations and custom Action

To begin with CDS View with BOPF CRUD operation, let's use a simple single table [don't want to perform with multiple tables with joins as this will make things little complex].

Below are the steps we are going to perform/show in this post-

1- Creating a table , 2-> Creating a CDS View with BOPF BO Enabled and with annotations with oData Publish, 3-> Testing the BO, 4-> Registering the oData and testing in gateway client, 5->creating a list report & object page fiori app,6-> Testing the basic CRUD on the fiori app,7->Adding a custom action to the BO, 8-> Testing the custom action on BO ,9->Enabling the custom action button on UI,10->Testing the custom action on the fiori app.

Created a simple table that stores the order details. The order status contains some domain fixed values.

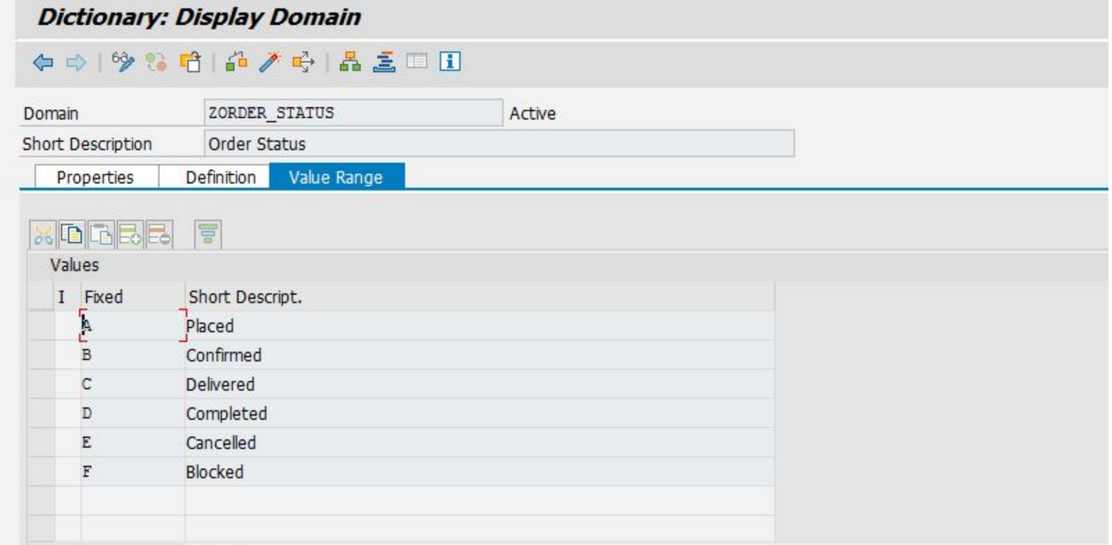


Transparent Table ZTAB_ORDER_HDR Active

Short Description Order Header

Field	Key	Ini...	Data element	Data Type	Length	Deci...	Coordinate	Short Description
CLIENT	☑	☑	MANDT	CLNT	3	0	0	Client
GUID	☑	☑	ZORDER_GUID	RAW	16	0	0	Order GUID
ID	□	□	ZORDER_ID	CHAR	10	0	0	Order ID
ORDER_DATE	□	□	ZORDER_DATE	DATS	8	0	0	Order Date
PRICE	□	□	ZORDER_PRICE	CURR	15	2	0	Order Value
CURRENCY	□	□	ZORDER_CURRENCY	CUKY	5	0	0	Order Currency
STATUS	□	□	ZORDER_STATUS	CHAR	1	0	0	Order Status

Domain fixed values- for the order status field.



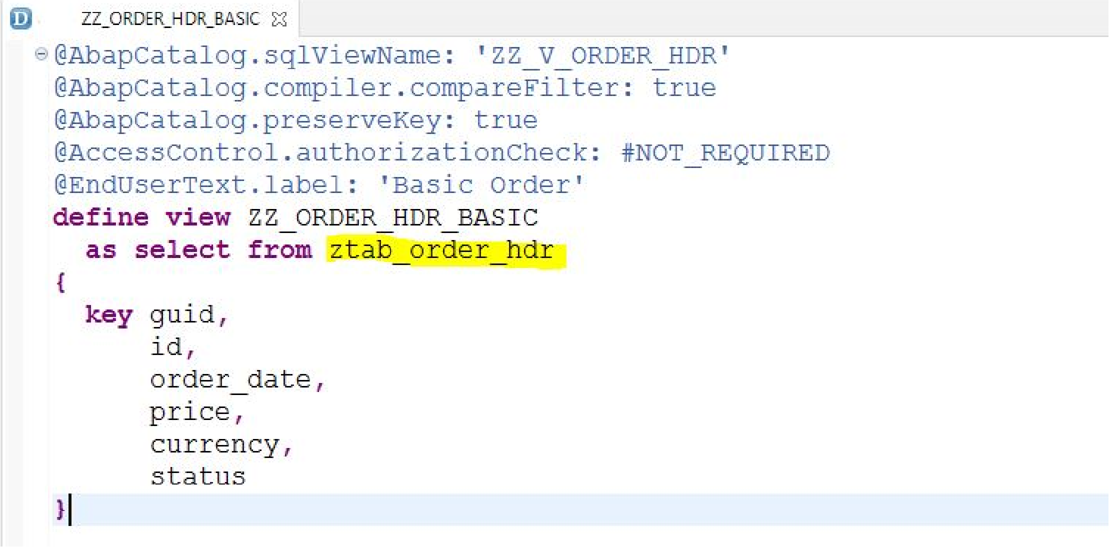
Dictionary: Display Domain

Domain ZORDER_STATUS Active

Short Description Order Status

I	Fixed	Short Descript.
	A	Placed
	B	Confirmed
	C	Delivered
	D	Completed
	E	Cancelled
	F	Blocked

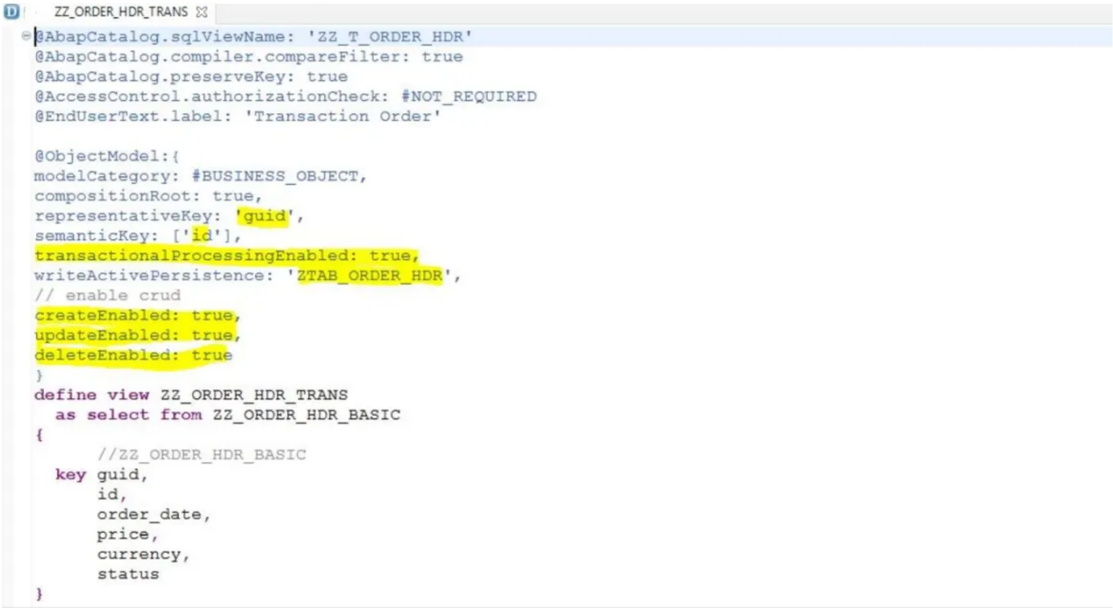
Let’s create a CDS view on top of the above DB table. Its quite simple.



```
@AbapCatalog.sqlViewName: 'ZZ_V_ORDER_HDR'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Basic Order'
define view ZZ_ORDER_HDR_BASIC
  as select from ztab_order_hdr
{
  key guid,
      id,
      order_date,
      price,
      currency,
      status
}
```

```
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: ‘Basic Order’
define view ZZ_ORDER_HDR_BASIC
as select from ztab_order_hdr
{
key guid,
id,
order_date,
price,
currency,
status
}
```

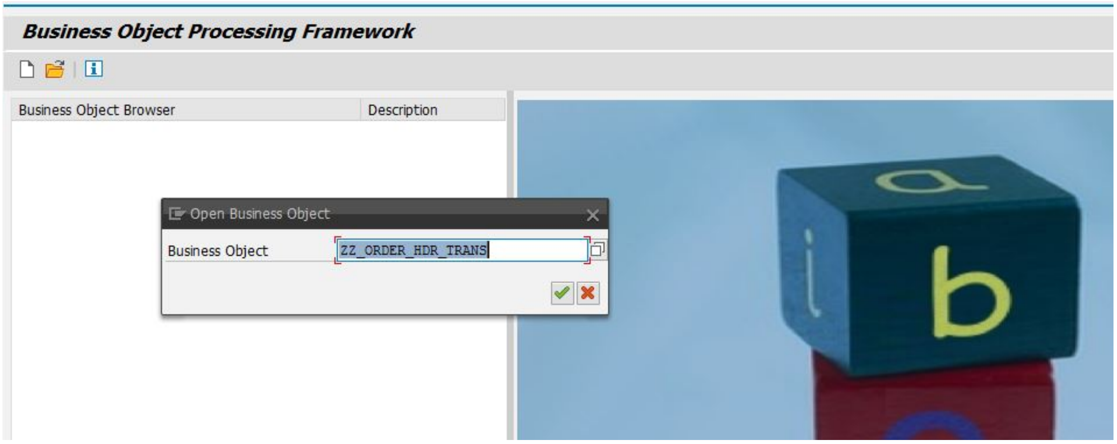
Now create another CDS view on top of the basic view created above with BO enabled.



```
@AbapCatalog.sqlViewName: 'ZZ_T_ORDER_HDR'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Transaction Order'

@ObjectModel:{
modelCategory: #BUSINESS_OBJECT,
compositionRoot: true,
representativeKey: 'guid',
semanticKey: ['id'],
transactionalProcessingEnabled: true,
writeActivePersistence: 'ZTAB_ORDER_HDR',
// enable crud
createEnabled: true,
updateEnabled: true,
deleteEnabled: true
}
define view ZZ_ORDER_HDR_TRANS
  as select from ZZ_ORDER_HDR_BASIC
{
      //ZZ_ORDER_HDR_BASIC
  key guid,
      id,
      order_date,
      price,
      currency,
      status
}
```

```
@AbapCatalog.sqlViewName: ‘ZZ_T_ORDER_HDR’
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: ‘Transaction Order’

@ObjectModel:{
modelCategory: #BUSINESS_OBJECT,
compositionRoot: true,
representativeKey: ‘guid’,
semanticKey: [‘id’],
transactionalProcessingEnabled: true,
writeActivePersistence: ‘ZTAB_ORDER_HDR’,
// enable crud
}
define view ZZ_ORDER_HDR_TRANS
as select from ZZ_ORDER_HDR_BASIC
{
//ZZ_ORDER_HDR_BASIC
key guid,
id,
order_date,
price,
currency,
status
}
```

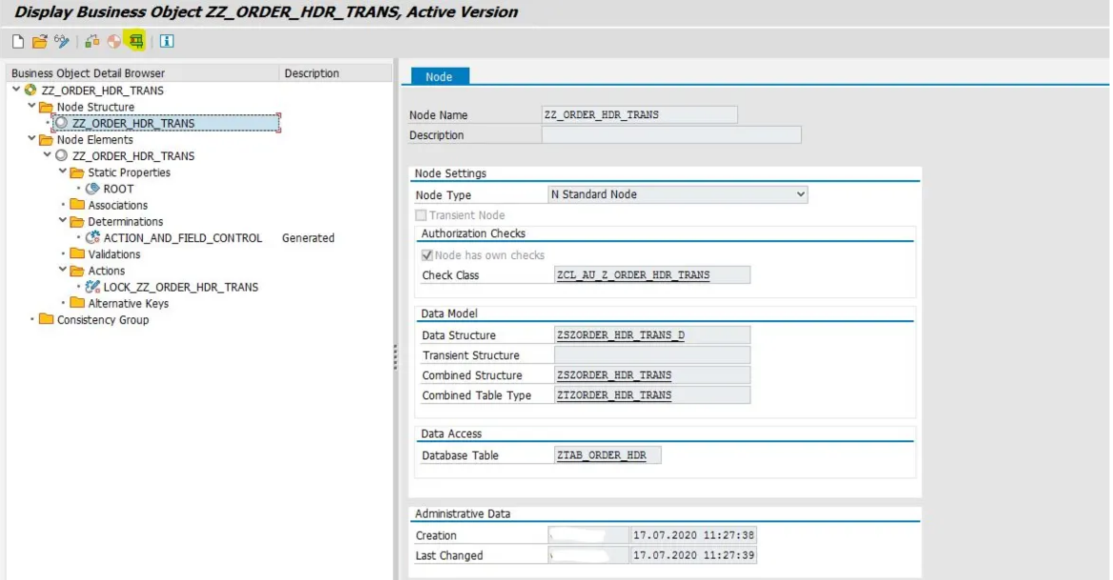
Execute Tcode- BOBX and Click on open BO button and provide the name of the CDS view which is the BO name also.



Business Object Processing Framework

Business Object: ZZ_ORDER_HDR_TRANS

In the node few objects like- table type,structure, constant interface and etc are already generated]. Now select the TEST button.



Display Business Object ZZ_ORDER_HDR_TRANS, Active Version

Node Name: ZZ_ORDER_HDR_TRANS
Description:

Node Settings
Node Type: N Standard Node
□ Transient Node

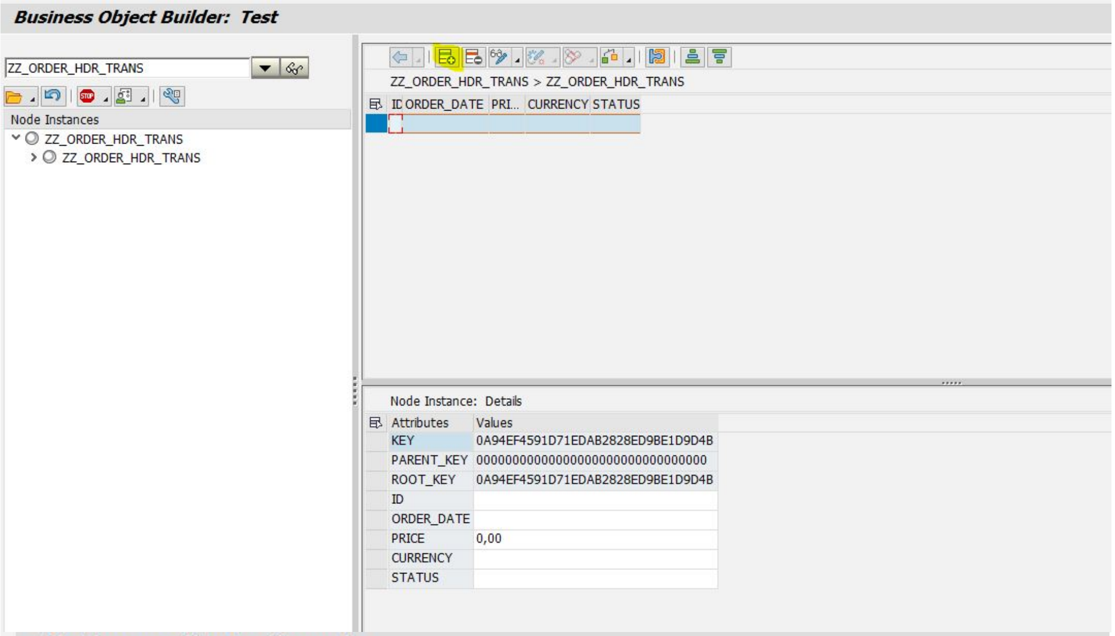
Authorization Checks
☑ Node has own checks
Check Class: ZCL_AU_Z_ORDER_HDR_TRANS

Data Model
Data Structure: ZSZORDER_HDR_TRANS_D
Transient Structure:
Combined Structure: ZSZORDER_HDR_TRANS
Combined Table Type: ZTZORDER_HDR_TRANS

Data Access
Database Table: ZTAB_ORDER_HDR

Administrative Data
Creation: [redacted] 17.07.2020 11:27:38
Last Changed: [redacted] 17.07.2020 11:27:39

Click on the "+" button. and Create some records and save.



Business Object Builder: Test

ZZ_ORDER_HDR_TRANS

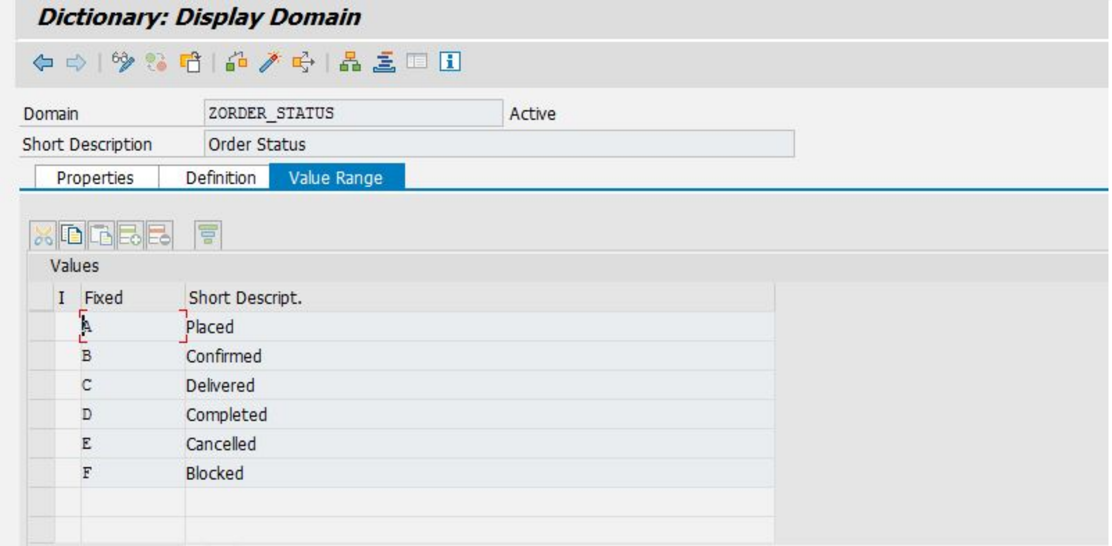
Node Instances
- ZZ_ORDER_HDR_TRANS
  - ZZ_ORDER_HDR_TRANS

ZZ_ORDER_HDR_TRANS > ZZ_ORDER_HDR_TRANS

ID	ORDER_DATE	PRI...	CURRENCY	STATUS

Node Instance: Details

Attributes	Values
KEY	0A94EF4591D71EDAB2828ED9BE1D9D4B
PARENT_KEY	00000000000000000000000000000000
ROOT_KEY	0A94EF4591D71EDAB2828ED9BE1D9D4B
ID	
ORDER_DATE	
PRICE	0,00
CURRENCY	
STATUS	



Dictionary: Display Domain

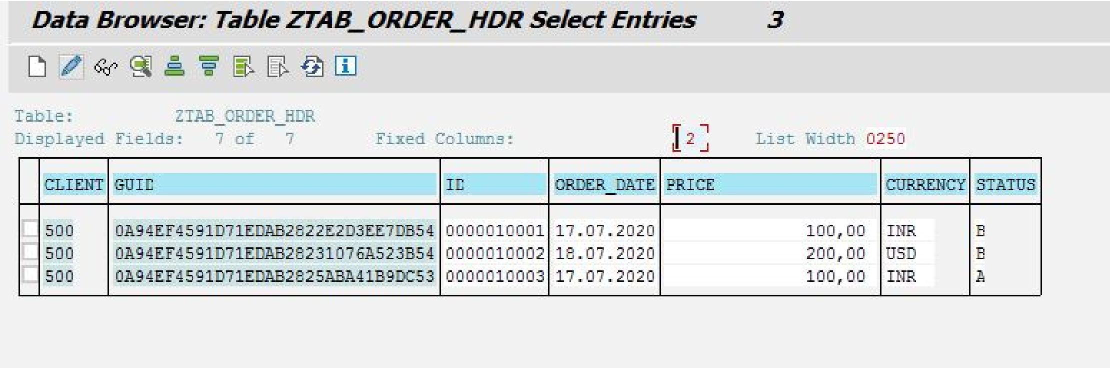
Domain ZORDER_STATUS Active

Short Description Order Status

Values

Fixed	Short Descript.
A	Placed
B	Confirmed
C	Delivered
D	Completed
E	Cancelled
F	Blocked

Check the table. We have few records now created in the DB table. The test of BO is successful.

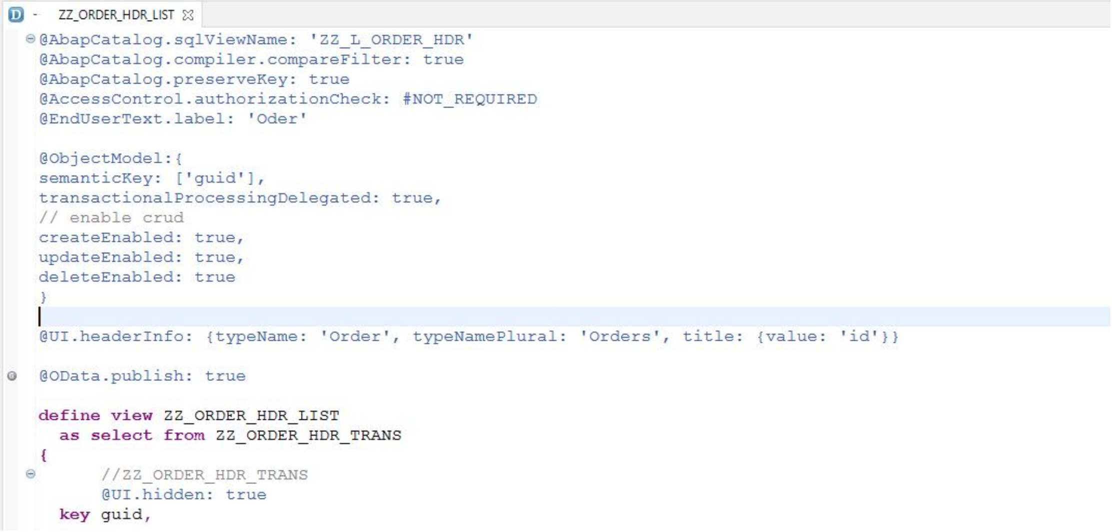


Data Browser: Table ZTAB_ORDER_HDR Select Entries 3

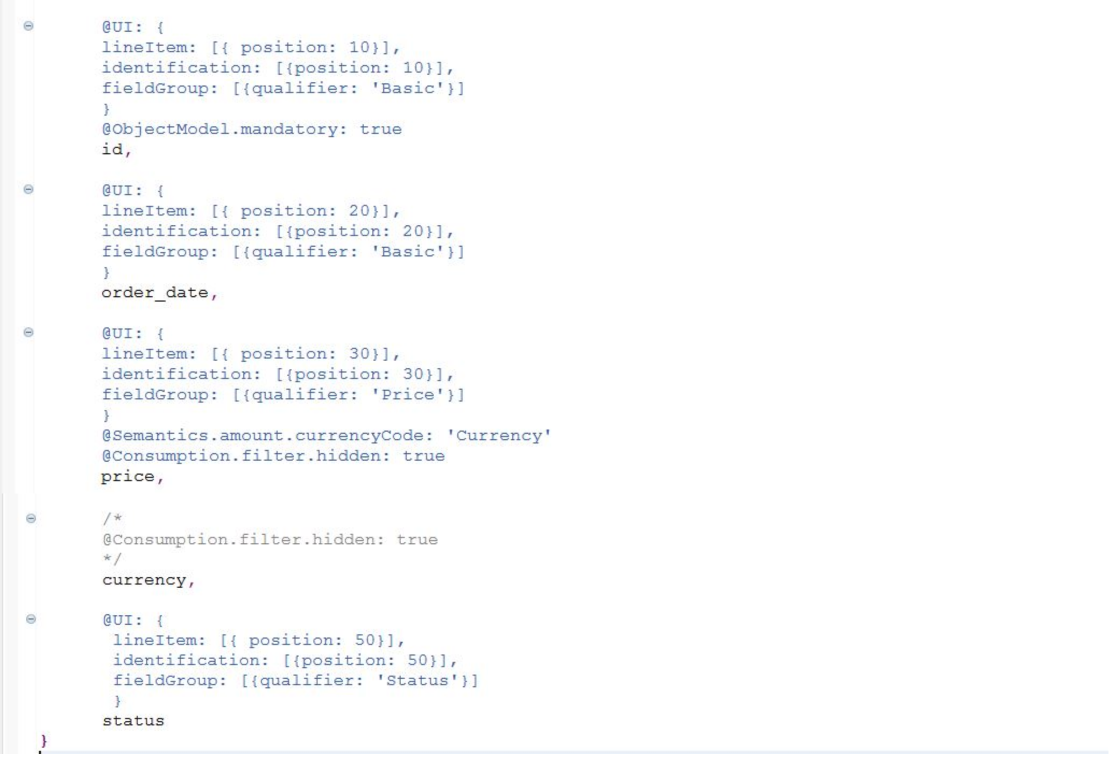
```
Table:           ZTAB_ORDER_HDR
Displayed Fields:   7 of   7       Fixed Columns:              2      List Width 0250
```

CLIENT	GUID	ID	ORDER_DATE	PRICE	CURRENCY	STATUS
500	0A94EF4591D71EDAB2822E2D3EE7DB54	0000010001	17.07.2020	100,00	INR	B
500	0A94EF4591D71EDAB28231076A523B54	0000010002	18.07.2020	200,00	USD	B
500	0A94EF4591D71EDAB2825ABA41B9DC53	0000010003	17.07.2020	100,00	INR	A

Now finally create another CDS View on top of the CDS view which is BO enabled and we provide some relevant annotations to the fields to design our LIST Report and Object Page. Make oData publish true for this view.



```
@AbapCatalog.sqlViewName: 'ZZ_L_ORDER_HDR'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Oder'

@ObjectModel:{
semanticKey: ['guid'],
transactionalProcessingDelegated: true,
// enable crud
createEnabled: true,
updateEnabled: true,
deleteEnabled: true
}

@UI.headerInfo: {typeName: 'Order', typeNamePlural: 'Orders', title: {value: 'id'}}

@OData.publish: true

define view ZZ_ORDER_HDR_LIST
  as select from ZZ_ORDER_HDR_TRANS
{
      //ZZ_ORDER_HDR_TRANS
      @UI.hidden: true
  key guid,
```



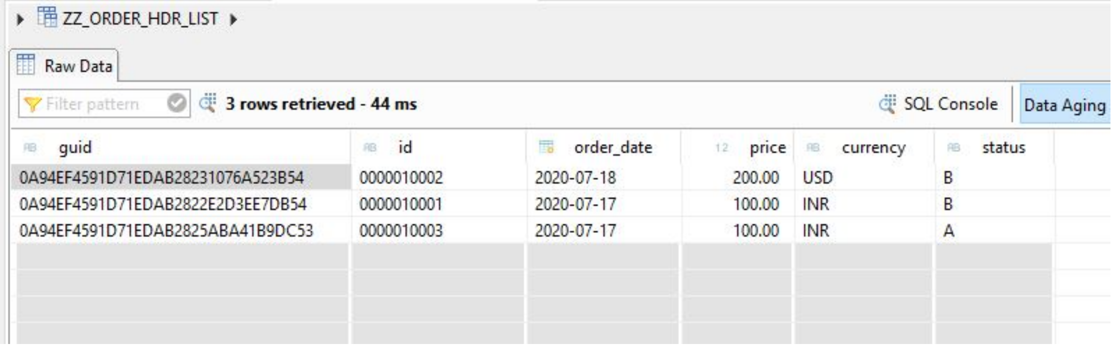
```
      @UI: {
      lineItem: [{ position: 10}],
      identification: [{position: 10}],
      fieldGroup: [{qualifier: 'Basic'}]
      }
      @ObjectModel.mandatory: true
      id,

      @UI: {
      lineItem: [{ position: 20}],
      identification: [{position: 20}],
      fieldGroup: [{qualifier: 'Basic'}]
      }
      order_date,

      @UI: {
      lineItem: [{ position: 30}],
      identification: [{position: 30}],
      fieldGroup: [{qualifier: 'Price'}]
      }
      @Semantics.amount.currencyCode: 'Currency'
      @Consumption.filter.hidden: true
      price,

      /*
      @Consumption.filter.hidden: true
      */
      currency,

      @UI: {
       lineItem: [{ position: 50}],
       identification: [{position: 50}],
       fieldGroup: [{qualifier: 'Status'}]
       }
      status
}
```

@AbapCatalog.sqlViewName: 'ZZ_L_ORDER_HDR'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Oder'

@ObjectModel:{
semanticKey: ['guid'],
transactionalProcessingDelegated: true,
// enable crud
createEnabled: true,

```
@UI.headerInfo: {typeName: ‘Order’, typeNamePlural: ‘Orders’, title: {value: ‘id’}}

@OData.publish: true

define view ZZ_ORDER_HDR_LIST
as select from ZZ_ORDER_HDR_TRANS
{
//ZZ_ORDER_HDR_TRANS
@UI.hidden: true
key guid,

@UI: {
lineItem: [{ position: 10}],
identification: [{position: 10}],
fieldGroup: [{qualifier: ‘Basic’}]
}
@ObjectModel.mandatory: true
id,

@UI: {
lineItem: [{ position: 20}],
identification: [{position: 20}],
fieldGroup: [{qualifier: ‘Basic’}]
}
order_date,

@UI: {
lineItem: [{ position: 30}],
identification: [{position: 30}],
fieldGroup: [{qualifier: ‘Price’}]
}
@Semantics.amount.currencyCode: ‘Currency’
@Consumption.filter.hidden: true
price,

/
@Consumption.filter.hidden: true
/
currency,

@UI: {
lineItem: [{ position: 50}],
identification: [{position: 50}],
fieldGroup: [{qualifier: ‘Status’}]
}
status
}
```

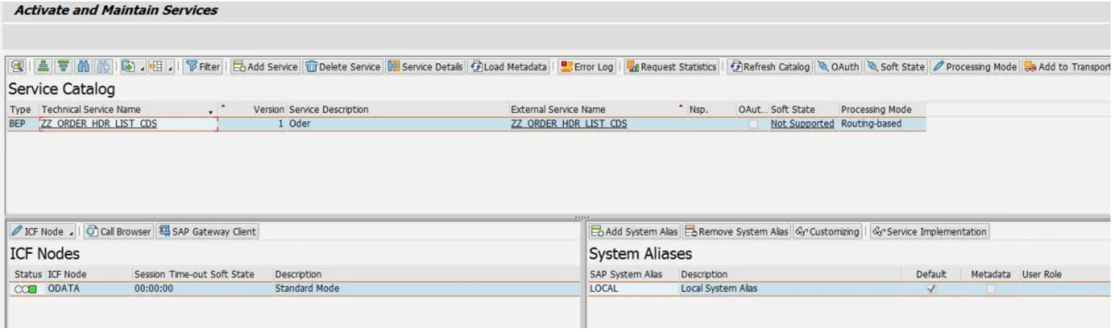
CDS View Data Preview Test.



3 rows retrieved - 44 ms

guid	id	order_date	price	currency	status
0A94EF4591D71EDAB28231076A523B54	0000010002	2020-07-18	200.00	USD	B
0A94EF4591D71EDAB2822E2D3EE7DB54	0000010001	2020-07-17	100.00	INR	B
0A94EF4591D71EDAB2825ABA41B9DC53	0000010003	2020-07-17	100.00	INR	A

Register the service ZZ_ORDER_HDR_LIST_CDS in the backend with Tcode- /IWFND/MAINT_SERVICE. Select the Gateway client to test the Odata Service.



Service Catalog

Type	Technical Service Name	Version	Service Description	External Service Name	Nsp.	OAut...	Soft State	Processing Mode
BEP	ZZ_ORDER_HDR_LIST_CDS	1	Oder	ZZ_ORDER_HDR_LIST_CDS		□	Not Supported	Routing-based

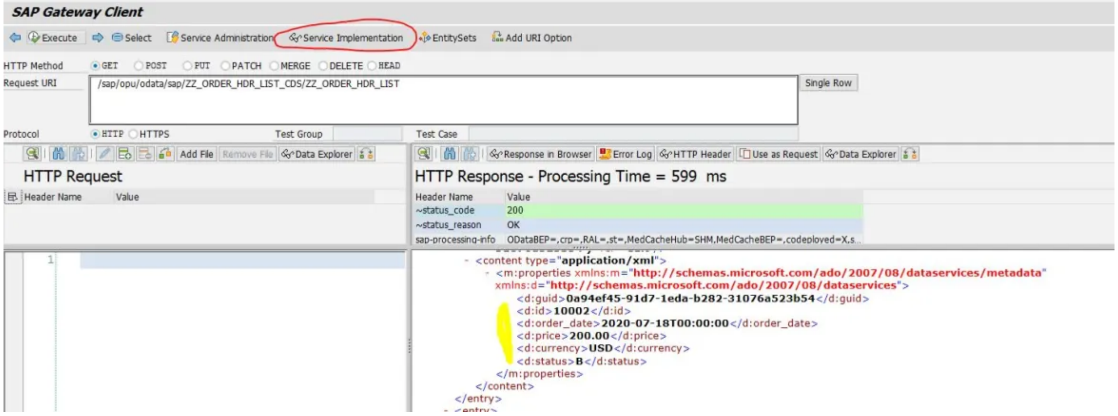
ICF Nodes

Status	ICF Node	Session Time-out	Soft State	Description
○○🟩	ODATA	00:00:00		Standard Mode

System Aliases

SAP System Alias	Description	Default	Metadata	User Role
LOCAL	Local System Alias	✓	□	

Test the entity set of the service with GET operation.[Other Operation like PUT,POST,DELETE can also be tested additionally] We Here we can see the Service Implementation details generated.



HTTP Method ● GET ○ POST ○ PUT ○ PATCH ○ MERGE ○ DELETE ○ HEAD

Request URI /sap/opu/odata/sap/ZZ_ORDER_HDR_LIST_CDS/ZZ_ORDER_HDR_LIST

Protocol ● HTTP ○ HTTPS

Test Group

Test Case

HTTP Request

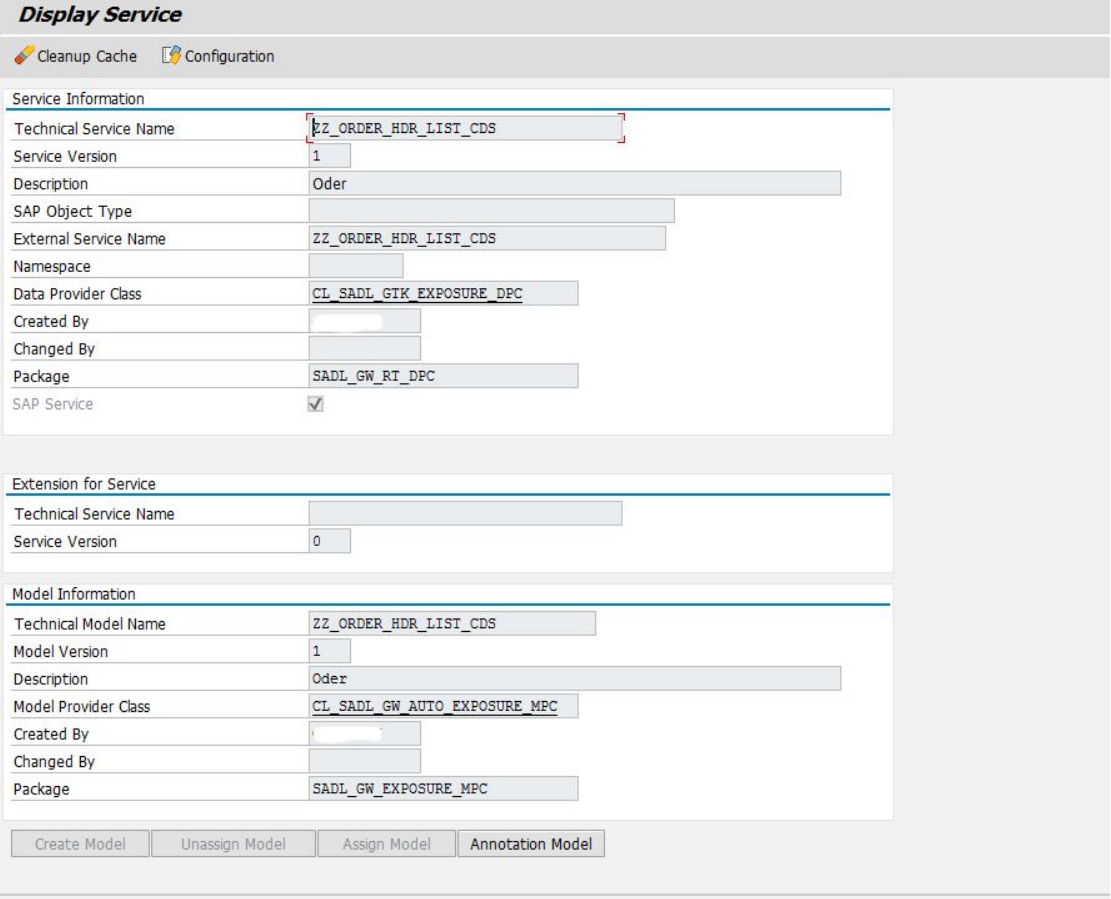
Header Name	Value

HTTP Response - Processing Time = 599 ms

Header Name	Value
~status_code	200
~status_reason	OK
sap-processing-info	ODataBEP=,crp=,RAL=,st=,MedCacheHub=SHM,MedCacheBEP=,codeployed=X,s...

```
  - <content type="application/xml">
    - <m:properties xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata"
      xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices">
        <d:guid>0a94ef45-91d7-1eda-b282-31076a523b54</d:guid>
        <d:id>10002</d:id>
        <d:order_date>2020-07-18T00:00:00</d:order_date>
        <d:price>200.00</d:price>
        <d:currency>USD</d:currency>
        <d:status>B</d:status>
      </m:properties>
    </content>
  </entry>
- <entry>
```

The Data Provider and Model Provider Class.



Display Service

Cleanup Cache | Configuration

Service Information

Technical Service Name	ZZ_ORDER_HDR_LIST_CDS
Service Version	1
Description	Oder
SAP Object Type	
External Service Name	ZZ_ORDER_HDR_LIST_CDS
Namespace	
Data Provider Class	CL_SADL_GTK_EXPOSURE_DPC
Created By	[redacted]
Changed By	
Package	SADL_GW_RT_DPC
SAP Service	☑

Extension for Service

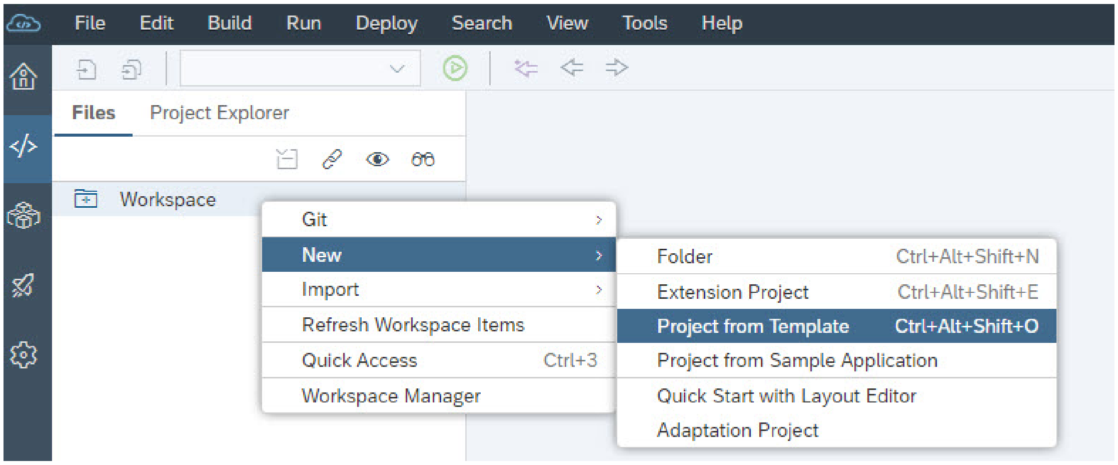
Technical Service Name	
Service Version	0

Model Information

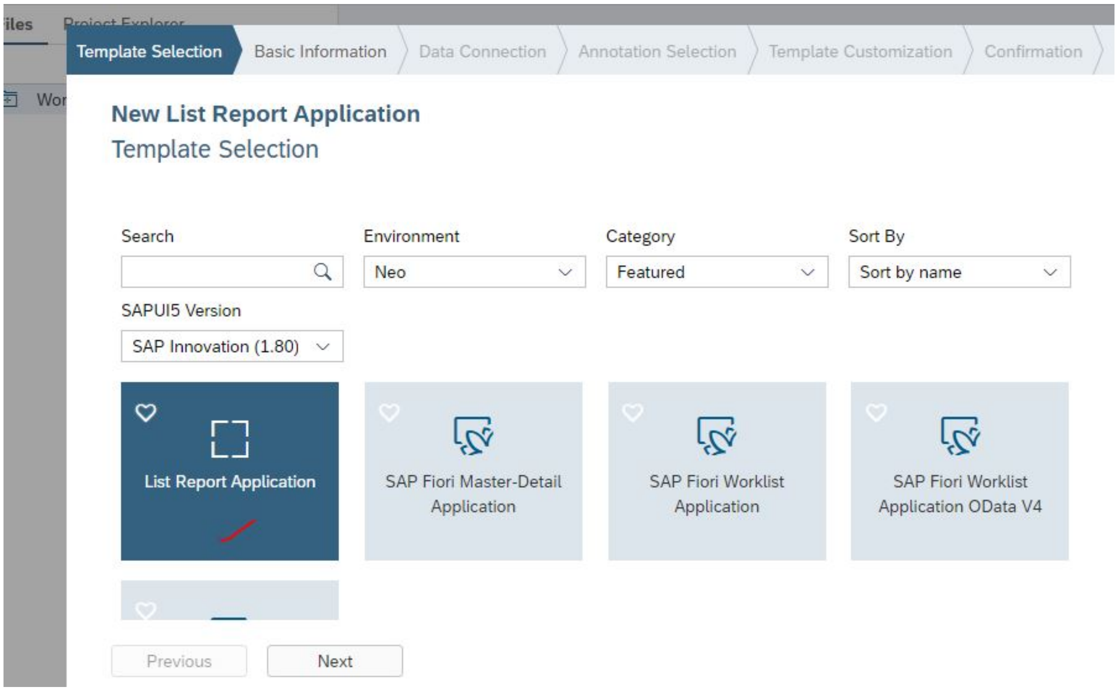
Technical Model Name	ZZ_ORDER_HDR_LIST_CDS
Model Version	1
Description	Oder
Model Provider Class	CL_SADL_GW_AUTO_EXPOSURE_MPC
Created By	[redacted]
Changed By	
Package	SADL_GW_EXPOSURE_MPC

Create Model | Unassign Model | Assign Model | Annotation Model

Now Time to create a Fiori App. Go to the WEB IDE and create a project from template.



Choose List report application.

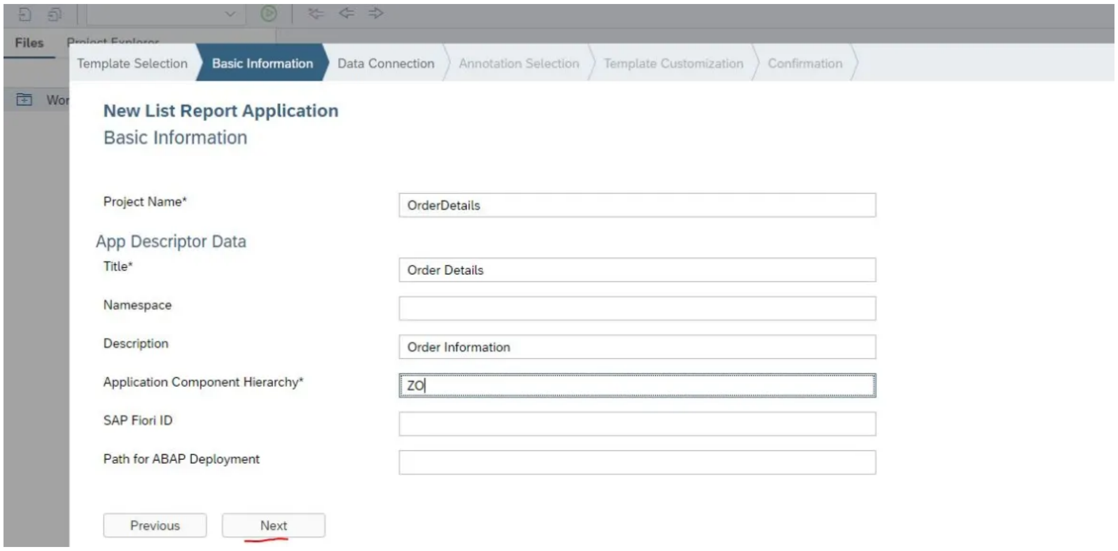


New List Report Application
Template Selection

Search:
Environment: Neo
Category: Featured
Sort By: Sort by name
SAPUI5 Version: SAP Innovation (1.80)

List Report Application
SAP Fiori Master-Detail Application
SAP Fiori Worklist Application
SAP Fiori Worklist Application OData V4
Previous
Next

Provide necessary details-



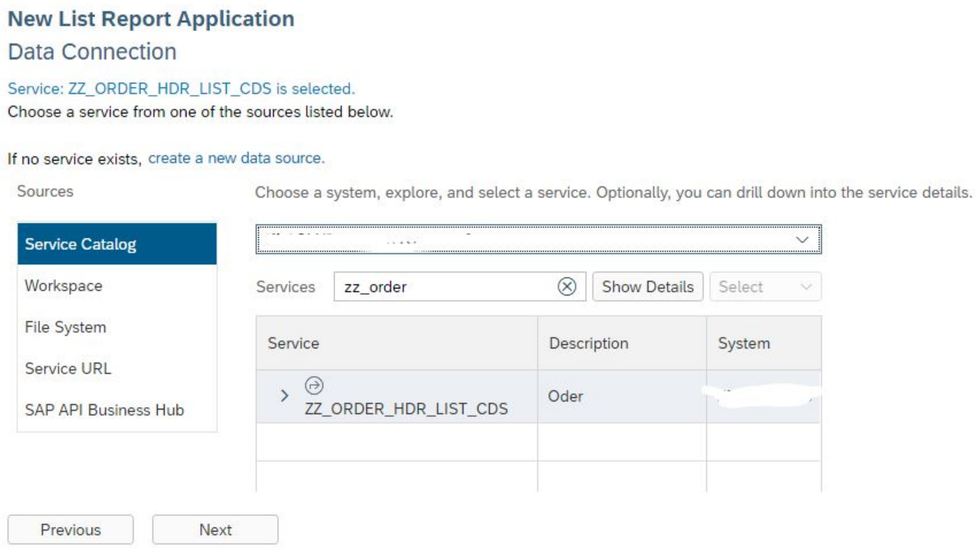
New List Report Application
Basic Information

Project Name*: OrderDetails

App Descriptor Data

Title*: Order Details
Namespace:
Description: Order Information
Application Component Hierarchy*: ZO
SAP Fiori ID:
Path for ABAP Deployment:

Provide the oData Service name.



New List Report Application

Data Connection

Service: ZZ_ORDER_HDR_LIST_CDS is selected.

Choose a service from one of the sources listed below.

If no service exists, create a new data source.

Sources

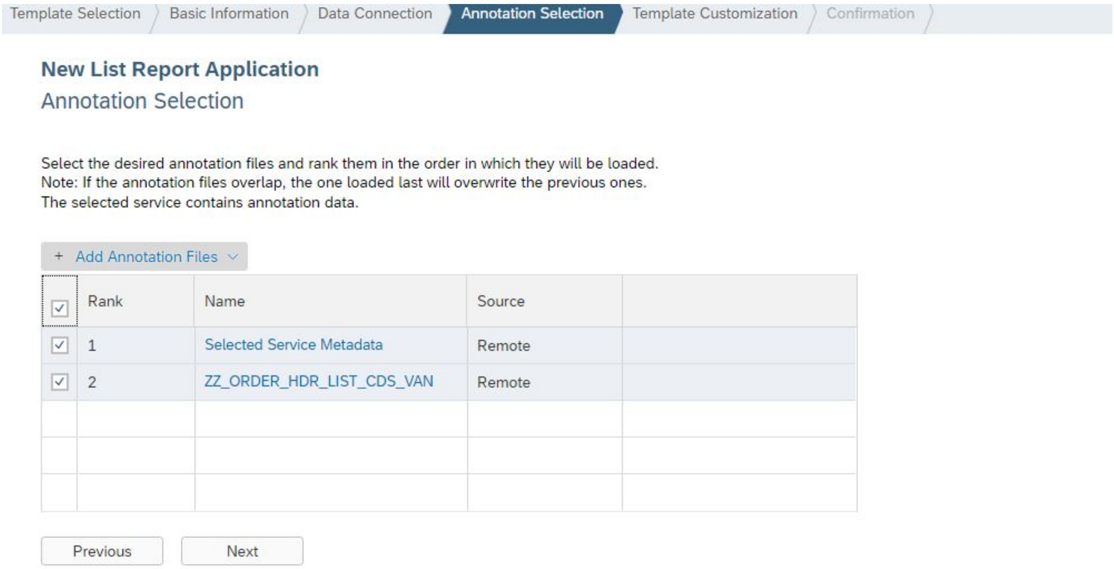
Choose a system, explore, and select a service. Optionally, you can drill down into the service details.

[redacted]

Services: zz_order

Service	Description	System
ZZ_ORDER_HDR_LIST_CDS	Oder	[redacted]

Select all-



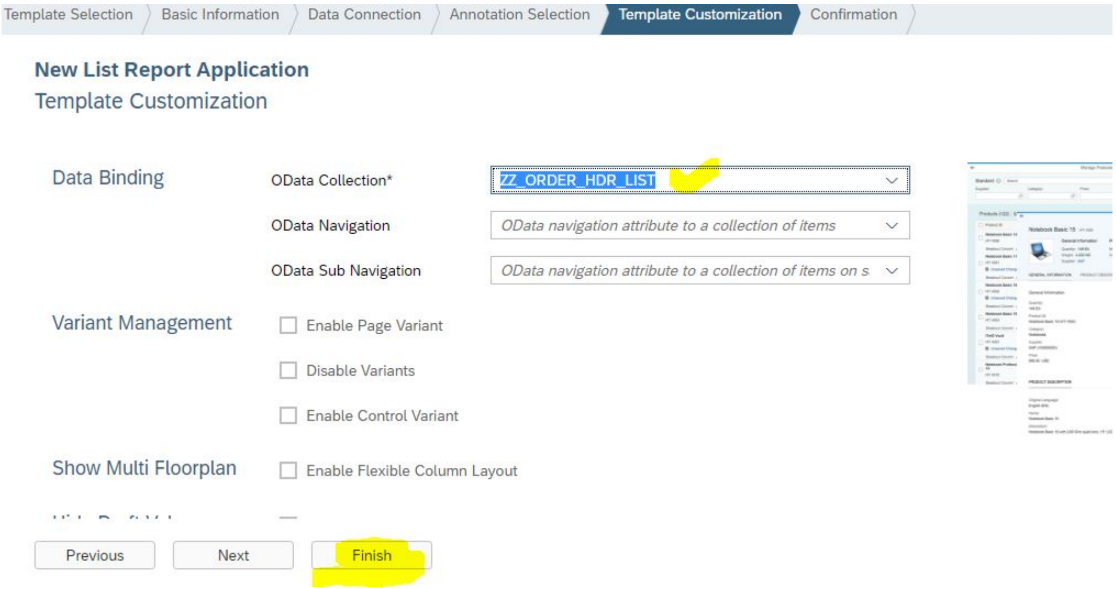
New List Report Application

Annotation Selection

Select the desired annotation files and rank them in the order in which they will be loaded.
Note: If the annotation files overlap, the one loaded last will overwrite the previous ones.
The selected service contains annotation data.

☑	Rank	Name	Source
☑	1	Selected Service Metadata	Remote
☑	2	ZZ_ORDER_HDR_LIST_CDS_VAN	Remote

Select oData Entity Set name and Finish.



New List Report Application

Template Customization

Data Binding

OData Collection*: ZZ_ORDER_HDR_LIST

OData Navigation: *OData navigation attribute to a collection of items*

OData Sub Navigation: *OData navigation attribute to a collection of items on s*

Variant Management

☐ Enable Page Variant

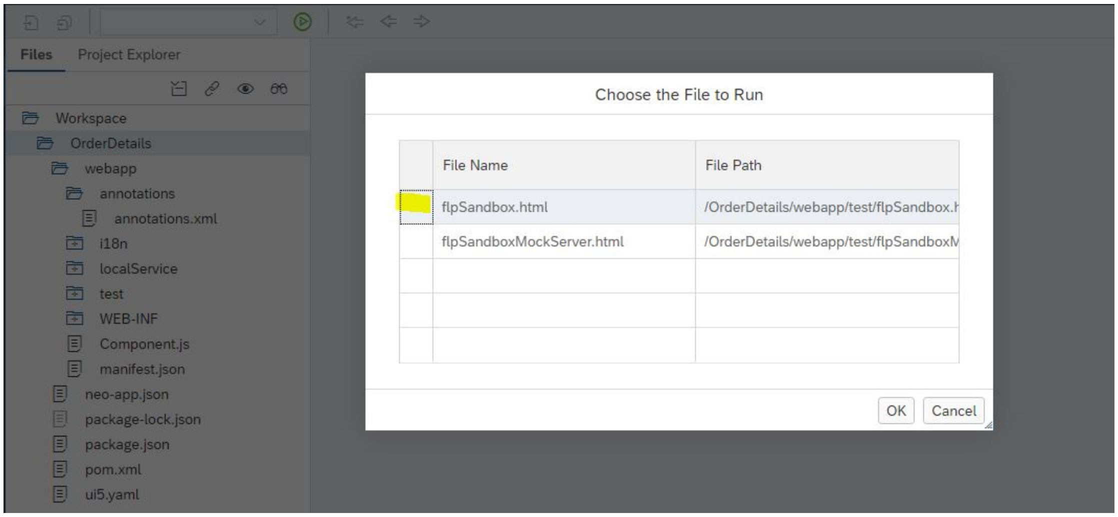
☐ Disable Variants

☐ Enable Control Variant

Show Multi Floorplan

☐ Enable Flexible Column Layout

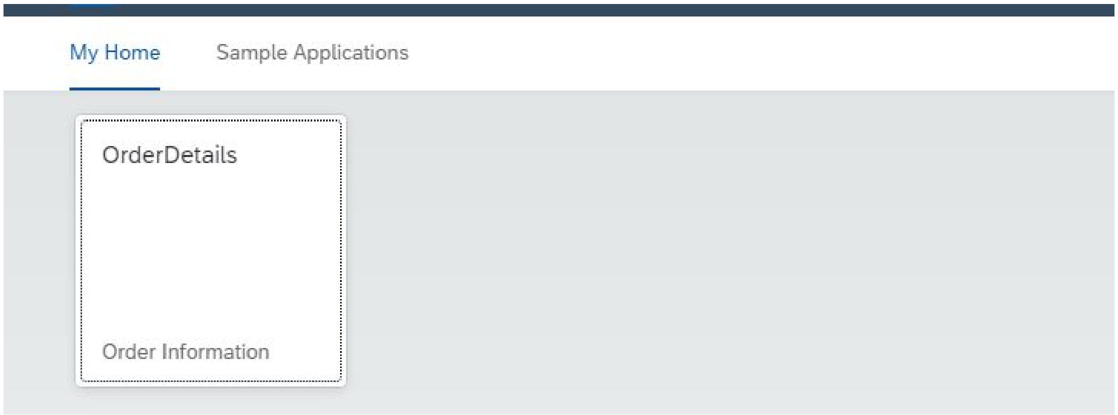
Run the app-



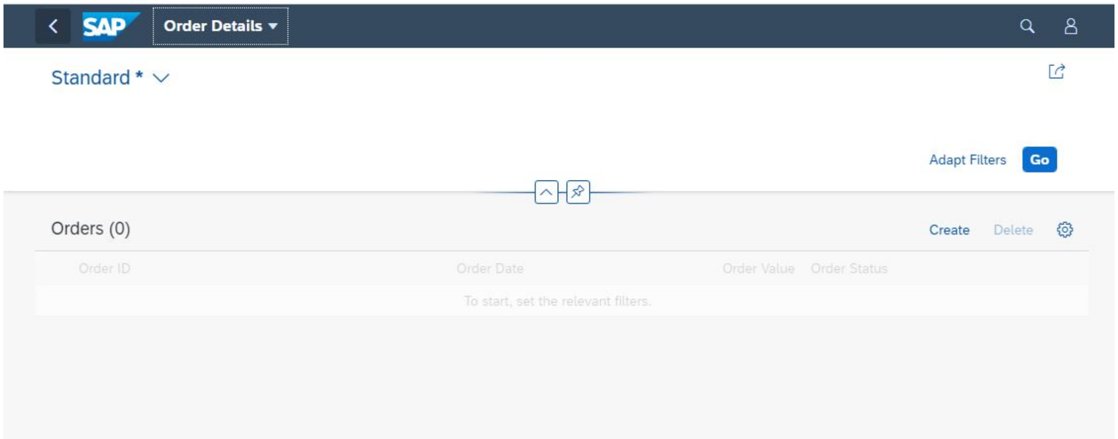
Choose the File to Run

	File Name	File Path
	flpSandbox.html	/OrderDetails/webapp/test/flpSandbox.h
	flpSandboxMockServer.html	/OrderDetails/webapp/test/flpSandboxM

Select to RUN the app-



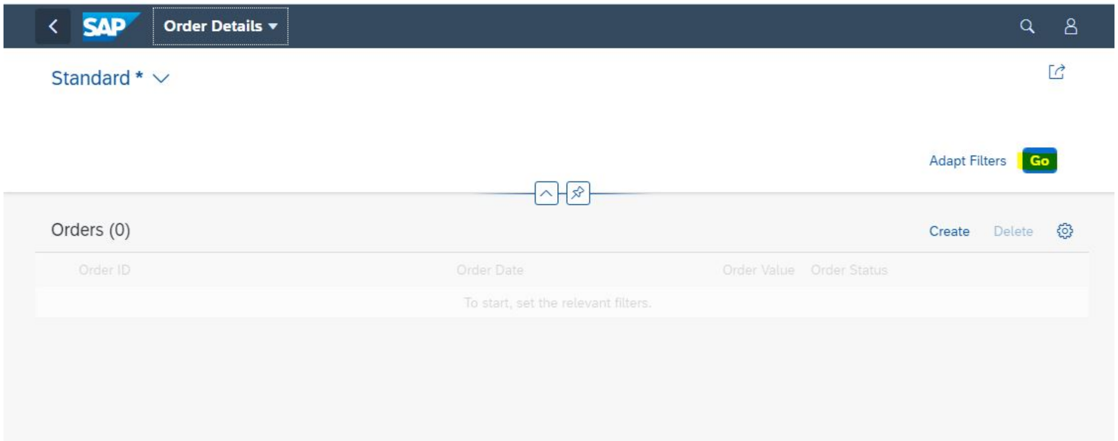
So here we have the search and list.



Order ID	Order Date	Order Value	Order Status

To start, set the relevant filters.

Select the GO button.



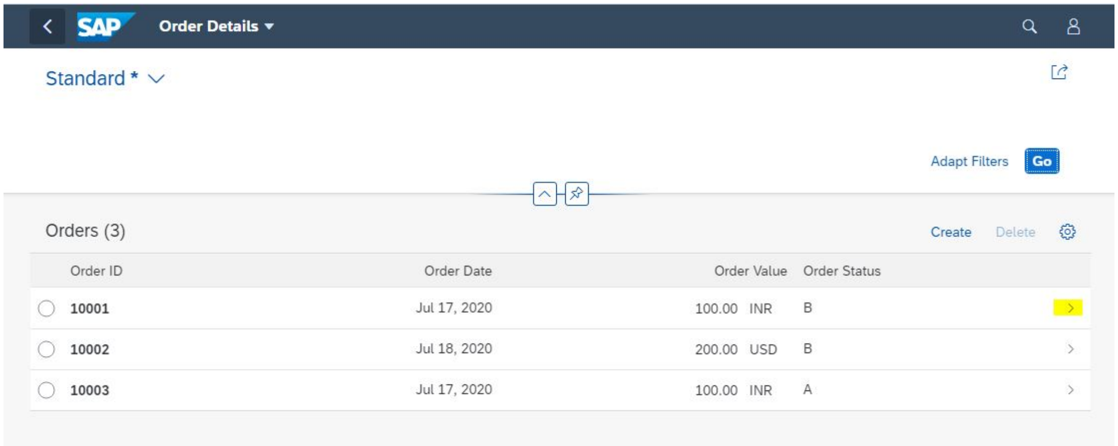
Order ID	Order Date	Order Value	Order Status

To start, set the relevant filters.

As we already created few DB records with our BO testing. Its gets all the records. So

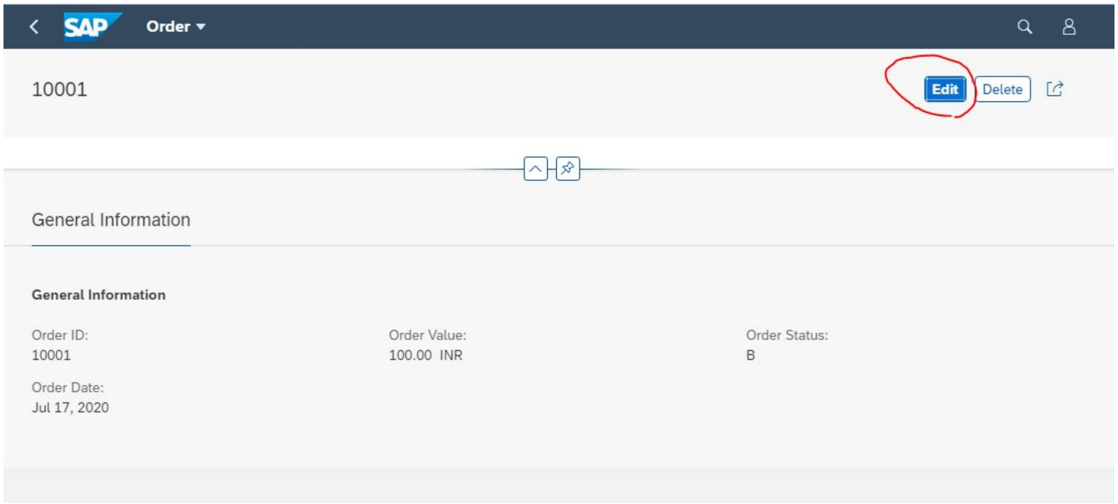
CRUD- R: Read Operation which is a HTTP GET method is now working.

Select a line to go to the Object Page.



	Order ID	Order Date	Order Value	Order Status	
○	**10001**	Jul 17, 2020	100.00 INR	B	>
○	**10002**	Jul 18, 2020	200.00 USD	B	>
○	**10003**	Jul 17, 2020	100.00 INR	A	>

Here we are in Object Page and let's EDIT it .



10001

General Information

General Information

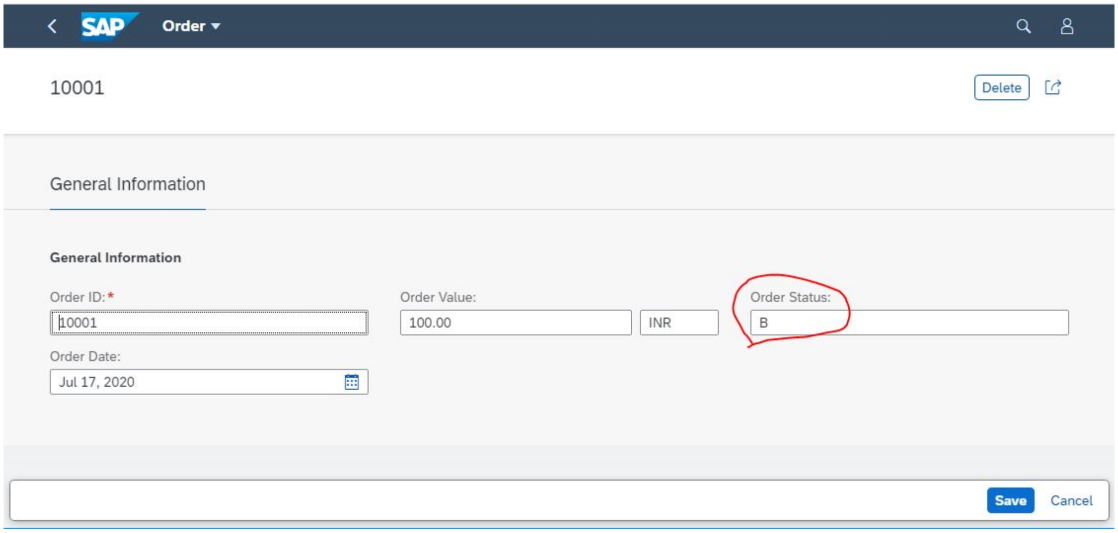
Order ID:
10001

Order Value:
100.00 INR

Order Status:
B

Order Date:
Jul 17, 2020

Now fields are editable .



10001

General Information

General Information

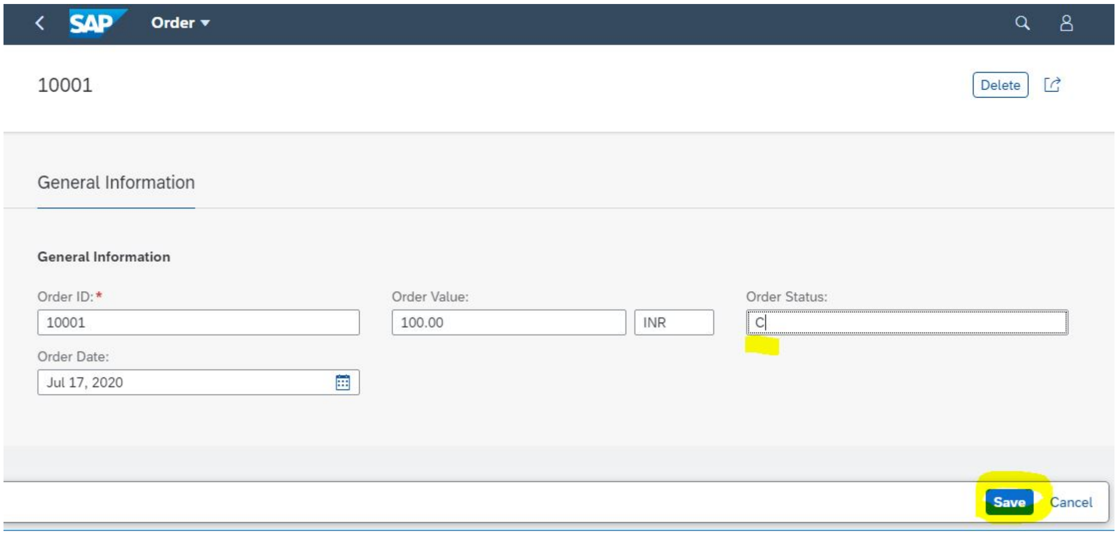
Order ID:* 10001

Order Value: 100.00 INR

Order Status: B

Order Date: Jul 17, 2020

Change the status value and save



10001

General Information

General Information

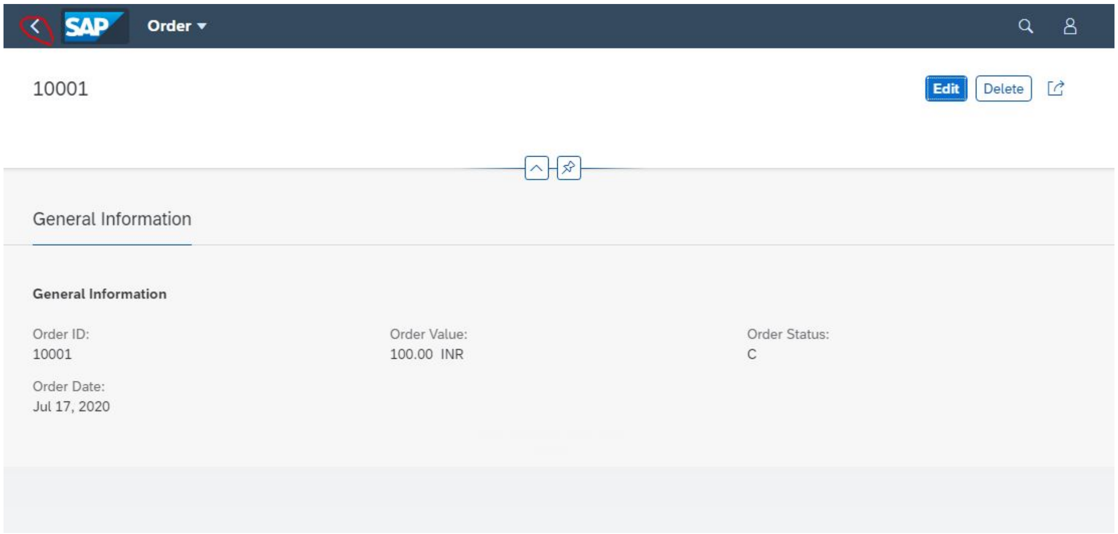
Order ID:* 10001

Order Value: 100.00 INR

Order Status: C

Order Date: Jul 17, 2020

The data is saved. The Update operation is done successfully. Go back.



10001

General Information

General Information

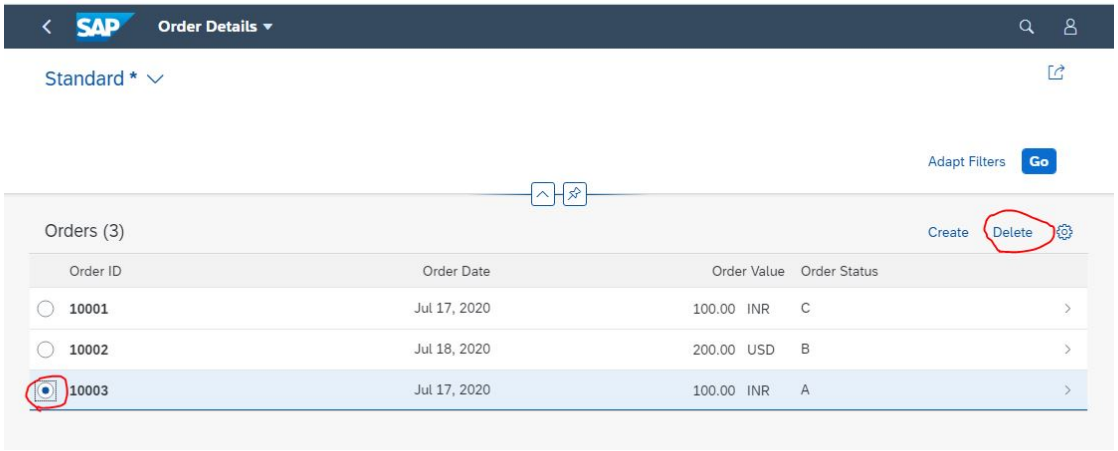
Order ID: 10001

Order Value: 100.00 INR

Order Status: C

Order Date: Jul 17, 2020

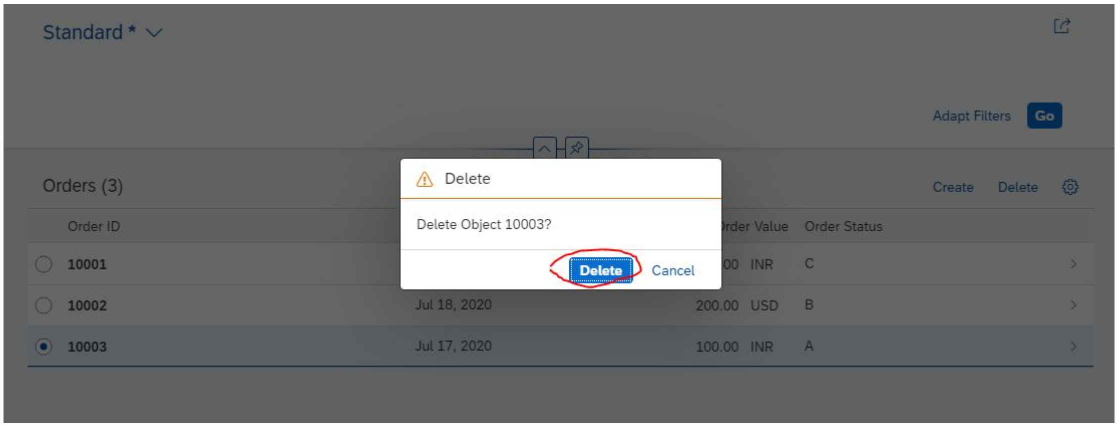
Select a line and Delete.



Orders (3)

	Order ID	Order Date	Order Value	Order Status
○	**10001**	Jul 17, 2020	100.00 INR	C
○	**10002**	Jul 18, 2020	200.00 USD	B
●	**10003**	Jul 17, 2020	100.00 INR	A

Confirm delete.



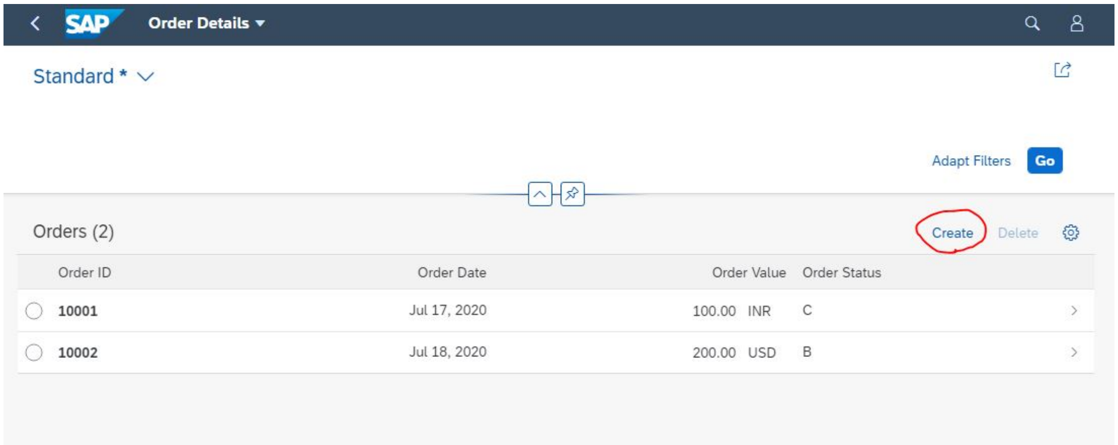
Orders (3)

⚠ Delete

Delete Object 10003?

	Order ID	Order Date	Order Value	Order Status
○	**10001**		00 INR	C
○	**10002**	Jul 18, 2020	200.00 USD	B
●	**10003**	Jul 17, 2020	100.00 INR	A

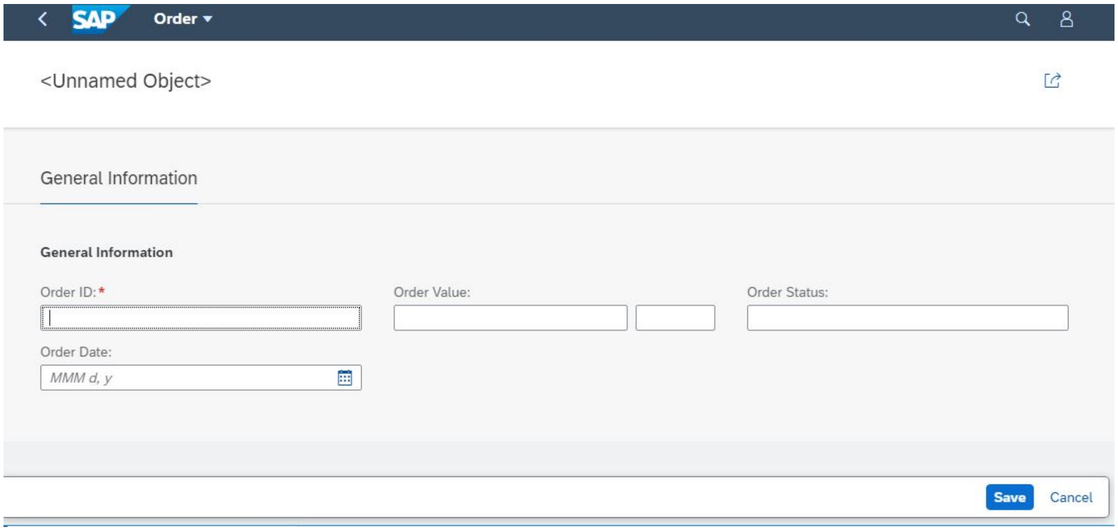
So DELETE operation is done successfully. Now try with CREATE operation.



Orders (2)

	Order ID	Order Date	Order Value	Order Status
○	10001	Jul 17, 2020	100.00 INR	C
○	10002	Jul 18, 2020	200.00 USD	B

The blank object page appears.



General Information

Order ID:*

Order Value:

Order Status:

Order Date: MMM d, y

Provide the values and save.

Go back.

So CREATE operation works fine.

Upto this our basic BOPF enabled CRUD operations works correctly in the App.

So next is to add a Action [to change the order status to complete] to the BO.

Open the BO with Tx- BOBX.

Provide the BO name.

Here i already added an action 'SET_STATUS_COMPLETE'. To add a new action go to the change mode and right click

Provide Action name and description and provide a class name and double click on the class to create the action handler class.

The class is created with the implemented interface. We have to implement the EXECUTE method as per the requirement. This is an empty method initially and we have to implement it.

The execute method signature details that will help to make implementation.

read the data you want to change and update the status and finally cal the UDPATE method. In Below we just set the status to D[Completed] .

The table entries.

Copy the GUID as we are going to test our action on this record.

Open the Tx- BOBT

Open the BO. Search with key.

Clock on "+" and provide the copied GUID of the record.

Let's put a breakpoint in the action handler EXECUTE method.

Sp we have the record details- the status is 'A' and if we apply the action it should change to D.

Apply the action-

The debugger details- Press F8 to finish the method execution.

Now the status is changed to D and now save.

The table is now updated. So the BO action works properly. Next thing is to make the BO action available on the App.

Let's make the STATUS field as readonly first.

Add the BO action to the CDS view with annotations so that it appears on the APP. Below we have enabled the action button on the list and the object page.

```
@AbapCatalog.sqlViewName: 'ZZ_L_ORDER_HDR'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
```

```
@ObjectModel:{
semanticKey: [‘guid’],
transactionalProcessingDelegated: true,
// enable crud
createEnabled: true,
updateEnabled: true,
deleteEnabled: true
}

@UI.headerInfo: {typeName: ‘Order’, typeNamePlural: ‘Orders’, title: {value: ‘id’}}

@OData.publish: true

define view ZZ_ORDER_HDR_LIST
as select from ZZ_ORDER_HDR_TRANS
{

@UI: {
lineItem: [{ type: #FOR_ACTION, position: 1,
dataAction: ‘BOPF:SET_STATUS_COMPLETE’,
label: ‘Set Complete’ }],
identification: [{type: #FOR_ACTION,
position: 1,
dataAction: ‘BOPF:SET_STATUS_COMPLETE’,
label: ‘Set Complete’ }]
}
//ZZ_ORDER_HDR_TRANS
@UI.hidden: true
key guid,

@UI: {
lineItem: [{ position: 10}],
identification: [{position: 10}],
fieldGroup: [{qualifier: ‘Basic’}]
}
@ObjectModel.mandatory: true
id,

@UI: {
lineItem: [{ position: 20}],
identification: [{position: 20}],
fieldGroup: [{qualifier: ‘Basic’}]
}
order_date,

@UI: {
lineItem: [{ position: 30}],
identification: [{position: 30}],
fieldGroup: [{qualifier: ‘Price’}]
}
/
@Consumption.filter.hidden: true
/
currency,

@UI: {
lineItem: [{ position: 50}],
identification: [{position: 50}],
fieldGroup: [{qualifier: 'Status'}]
}
@ObjectModel.readOnly: true
status
}
```

Run the app and the Action Button appears. Select a line.

The button now enabled and now press the button to change the status.

The status changed.

On the table also it is reflected.

Select a line and go to the object page.

Press the action button.

Seems the data is not getting refresh on the object page after the action. Go back.

Now it appears correctly here.

On the DB also it is changed.

To start with a simple cds view with BOPF operations, this is ok but there are still we can do a lot of things – like data is not getting refresh in the object page after action. We can also show the status text instead of single character value.

Also during the creation the status may be filled with the initial status value. So will try to do these things in the upcoming post.

See the post Fiori Elemnet- CDS View with BOPF CRUD(2)   to set the order status with BOPF determination.