Former Member

September 20, 2016 | 7 minute read

Create Fiori app using CDS with BOPF- For beginners Part 1

This blog is for developers who wants to get started with the UI5/Fiori + OData + CDS + BOPF.

Prerequisite:

1) Concepts of Core Data Services(CDS)

2) Basic Understanding of OData service

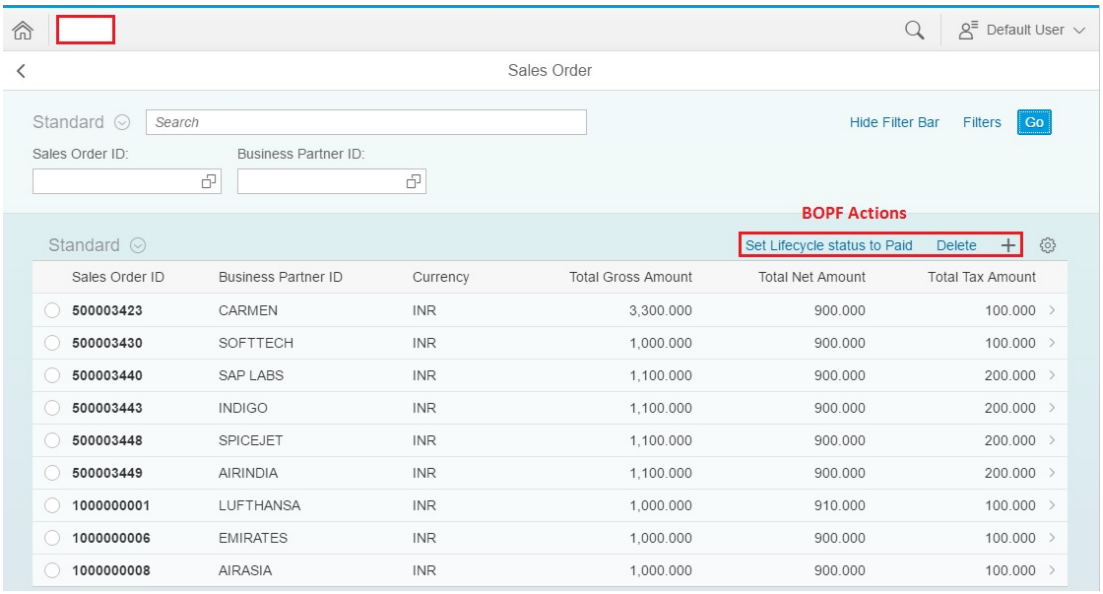
3) Concepts of BOPF

4) Conceptual knowledge of Fiori(Smart Template)

The Fiori/Ui5 is for the UI part, the CDS is for the data retrieval(code push down) while the BOPF is for handling DB activities.

Technically in the S4 Hana world, with the Code-Push down paradigm the intense business logic should happen in DB layer rather than in ABAP Application server. This blog is for pure test demo using CDS + BOPF to display the Fiori App. The performance optimization is not the intention here, but rather on how to use CDS + BOPF to get started with in S4 Hana Cloud.

Am using Web IDE environment with cloud connector to display the Fiori app. Let me quickly show how does the app look like.



The **+** and **Delete** buttons are handled by BOPF framework while the other action is manually created in BOPF. The Business logic of this other action shall be done in an BOPF action class. We shall get into the details later on.

Lets get started with the development.

Initial set up required

1) You need HANA studio or Eclipse tool as we need to use ADT(ABAP Development Tool)

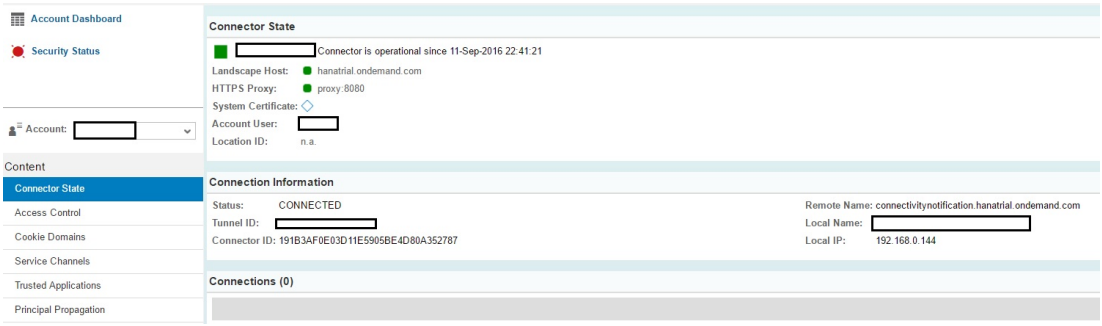
2) For hosting the app, Web IDE has been used. I recommend checking the link on how to install and set up the Web IDE trial and how to set up the Cloud connector.

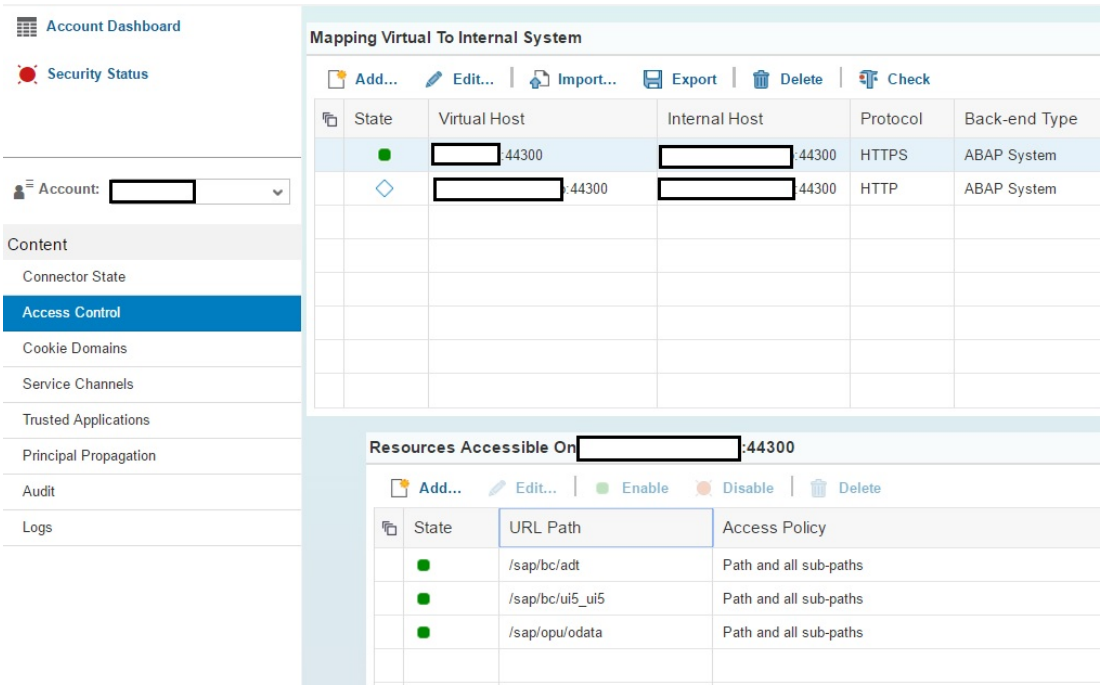
3) System is S4 HANA ON PREMISE 1.0 with ABAP 7.50

For those who are using Hana account trial, the link in step 2 will take care of the web IDE including the cloud connector. Make sure the cloud connector is connected to the backend system as per the set up in step 2.

If the Cloud Connector set up is done, it should look something like this:

Cloud connector

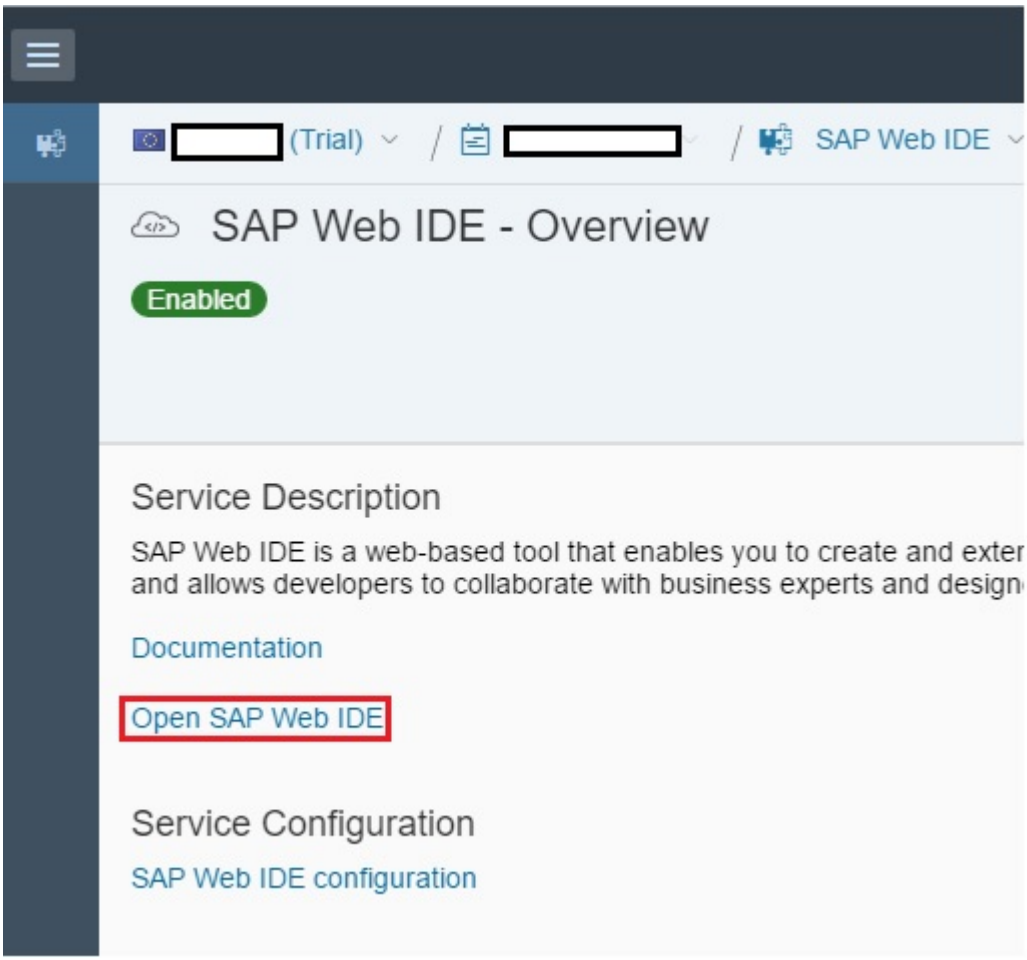




Web IDE

After logging into HANA account trial and go to services and click on Web IDE.

Open Web IDE now.



We shall be creating an sales order app as an demo example.

Basic steps involved in creating this app are :

- Create 2 Consumption CDS views (Sales Order header & Sales Order Items)
- Use annotation (Consumption view) for generating OData service
- Use annotations (Basic views) for generating BOPF objects
- Use Fiori Smart Template to display Sales order application

Technically at a higher level :

– The Fiori app shall consume the OData

– The OData has data source as CDS views (ie consumption CDS)

– The BOPF takes care of table CRUD operations. Here the BOPF objects are generated from the CDS views via annotations.

Lets start with the CDS views.

Basically in the CDS, there are consumption CDS, Basic CDS and Composite CDS. Consumption CDS are exposed to the UI. The intention is not to get into CDS details here, but to give a quick information on CDS.

We have 2 consumption CDS view's. And the Header Consumption view shall be used by Fiori. For simplicity sake, the naming convention shall be as :
Consumption CDS with *_C_*
Basic CDS with *_I_*

1) Header Consumption ZDEMO_C_SALESORDER

1.1) The SO header consumption view utilize the Basic CDS view ZDEMO_I_SALESORDER

1.2) The Basic view use a Header Table "ZPROTO_SO_A". This shall form the BOPF root table for SO Header

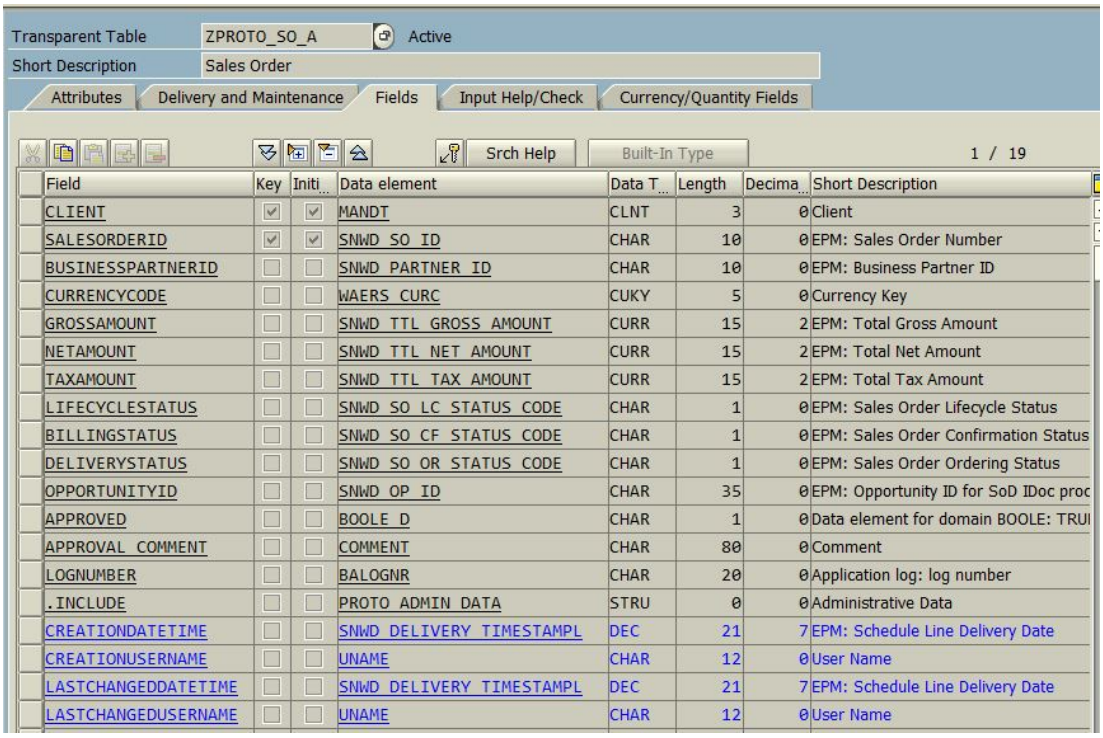
2) Item consumption view is ZDEMO_C_SALESORDER_ITEM

2.1) The Item consumption view use Basic CDS view ZDEMO_I_SALESORDER_ITEM

2.2) The Basic view use the Item table ZPROTO_SOI_A which shall form the BOPF item node table

First prepare the table structure for SO header and SO item respectively.

SO header table ZPROTO_SO_A



Transparent Table ZPROTO_SO_A Active

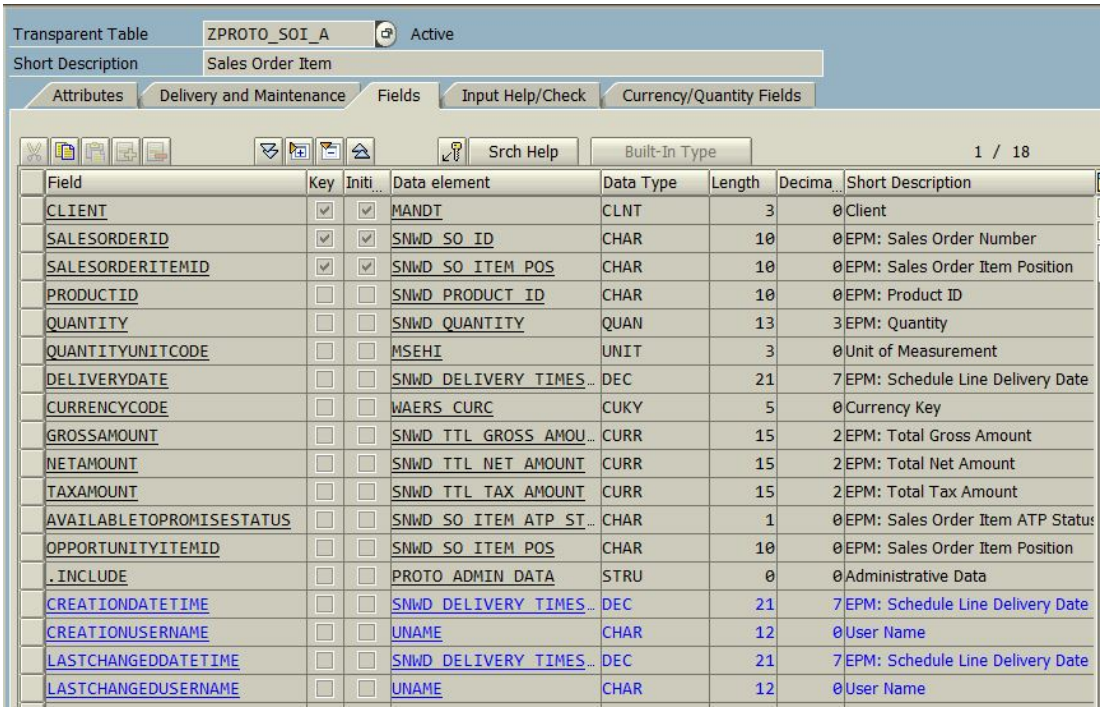
Short Description Sales Order

Attributes | Delivery and Maintenance | Fields | Input Help/Check | Currency/Quantity Fields

Srch Help | Built-In Type | 1 / 19

Field	Key	Initi...	Data element	Data T...	Length	Decima...	Short Description
CLIENT	✓	✓	MANDT	CLNT	3	0	Client
SALESORDERID	✓	✓	SNWD_SO_ID	CHAR	10	0	EPM: Sales Order Number
BUSINESSPARTNERID			SNWD_PARTNER_ID	CHAR	10	0	EPM: Business Partner ID
CURRENCYCODE			WAERS_CURC	CUKY	5	0	Currency Key
GROSSAMOUNT			SNWD_TTL_GROSS_AMOUNT	CURR	15	2	EPM: Total Gross Amount
NETAMOUNT			SNWD_TTL_NET_AMOUNT	CURR	15	2	EPM: Total Net Amount
TAXAMOUNT			SNWD_TTL_TAX_AMOUNT	CURR	15	2	EPM: Total Tax Amount
LIFECYCLESTATUS			SNWD_SO_LC_STATUS_CODE	CHAR	1	0	EPM: Sales Order Lifecycle Status
BILLINGSTATUS			SNWD_SO_CF_STATUS_CODE	CHAR	1	0	EPM: Sales Order Confirmation Status
DELIVERYSTATUS			SNWD_SO_OR_STATUS_CODE	CHAR	1	0	EPM: Sales Order Ordering Status
OPPORTUNITYID			SNWD_OP_ID	CHAR	35	0	EPM: Opportunity ID for SoD IDoc proc
APPROVED			BOOLE_D	CHAR	1	0	Data element for domain BOOLE: TRUI
APPROVAL_COMMENT			COMMENT	CHAR	80	0	Comment
LOGNUMBER			BALOGNR	CHAR	20	0	Application log: log number
.INCLUDE			PROTO_ADMIN_DATA	STRU	0	0	Administrative Data
CREATIONDATETIME			SNWD_DELIVERY_TIMESTAMPL	DEC	21	7	EPM: Schedule Line Delivery Date
CREATIONUSERNAME			UNAME	CHAR	12	0	User Name
LASTCHANGEDDATETIME			SNWD_DELIVERY_TIMESTAMPL	DEC	21	7	EPM: Schedule Line Delivery Date
LASTCHANGEDUSERNAME			UNAME	CHAR	12	0	User Name

SO Item table ZPROTO_SOI_A



Transparent Table ZPROTO_SOI_A Active

Short Description Sales Order Item

Attributes | Delivery and Maintenance | Fields | Input Help/Check | Currency/Quantity Fields

Srch Help | Built-In Type | 1 / 18

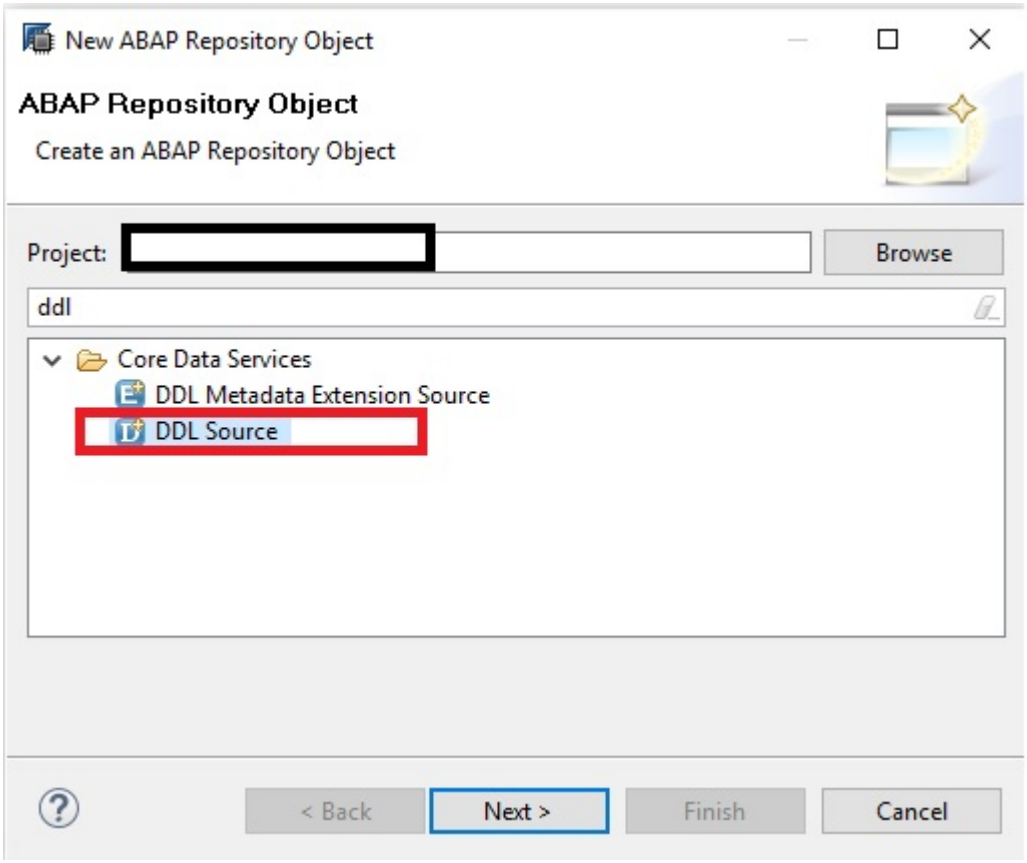
Field	Key	Initi...	Data element	Data Type	Length	Decima...	Short Description
CLIENT	✓	✓	MANDT	CLNT	3	0	Client
SALESORDERID	✓	✓	SNWD_SO_ID	CHAR	10	0	EPM: Sales Order Number
SALESORDERITEMID	✓	✓	SNWD_SO_ITEM_POS	CHAR	10	0	EPM: Sales Order Item Position
PRODUCTID			SNWD_PRODUCT_ID	CHAR	10	0	EPM: Product ID
QUANTITY			SNWD_QUANTITY	QUAN	13	3	EPM: Quantity
QUANTITYUNITCODE			MSEHI	UNIT	3	0	Unit of Measurement
DELIVERYDATE			SNWD_DELIVERY_TIMES...	DEC	21	7	EPM: Schedule Line Delivery Date
CURRENCYCODE			WAERS_CURC	CUKY	5	0	Currency Key
GROSSAMOUNT			SNWD_TTL_GROSS_AMOU...	CURR	15	2	EPM: Total Gross Amount
NETAMOUNT			SNWD_TTL_NET_AMOUNT	CURR	15	2	EPM: Total Net Amount
TAXAMOUNT			SNWD_TTL_TAX_AMOUNT	CURR	15	2	EPM: Total Tax Amount
AVAILABLETOPROMISESTATUS			SNWD_SO_ITEM_ATP_ST...	CHAR	1	0	EPM: Sales Order Item ATP Status
OPPORTUNITYITEMID			SNWD_SO_ITEM_POS	CHAR	10	0	EPM: Sales Order Item Position
.INCLUDE			PROTO_ADMIN_DATA	STRU	0	0	Administrative Data
CREATIONDATETIME			SNWD_DELIVERY_TIMES...	DEC	21	7	EPM: Schedule Line Delivery Date
CREATIONUSERNAME			UNAME	CHAR	12	0	User Name
LASTCHANGEDDATETIME			SNWD_DELIVERY_TIMES...	DEC	21	7	EPM: Schedule Line Delivery Date
LASTCHANGEDUSERNAME			UNAME	CHAR	12	0	User Name

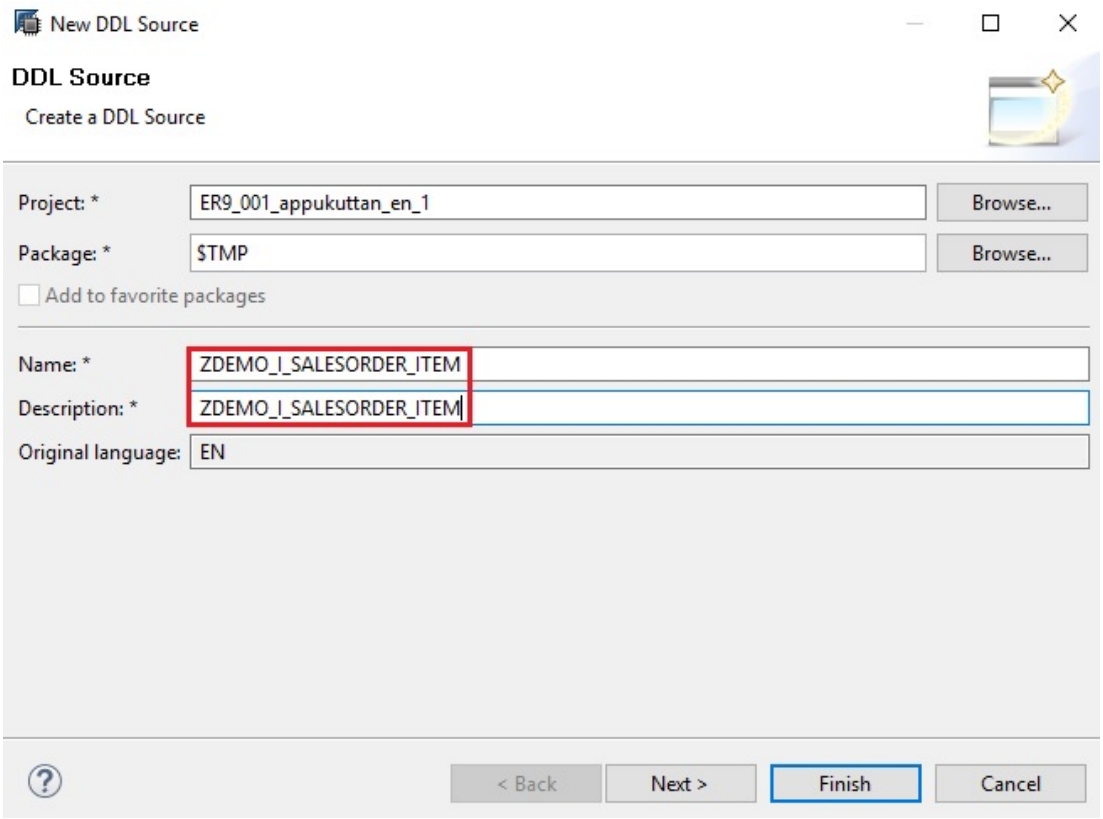
Now, we can use these tables in Basic CDS views.

Lets start creating with Basic Item CDS view

Basic SO Item CDS

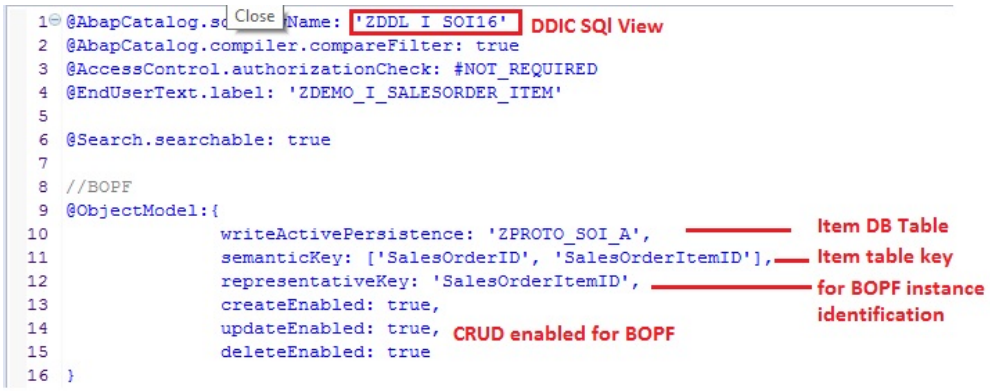
For creating CDS view, in the HANA studio, go to ABAP perspective. Choose your system and create a Package underneath. Right click on the package and choose New -> Other repository object.





Give the name of the view and click on Finish.

The CDS + annotations used are:



```
//CDS View
define view Zdemo_I_Salesorder_Item Item CDS

  as select from zproto_soi_a as Item

  //Association to Header                    Header Association
  association [1..1] to Zdemo_I_Salesorder as _SalesOrder on $projection.SalesOrderID = _SalesOrder.SalesOrderID

{
      //Sales order
      @Search.defaultSearchElement: true
      @ObjectModel.readOnly: true
  key Item.salesorderid             as SalesOrderID,

      //Sales order item
      @Search.defaultSearchElement: true
  key Item.salesorderitemid         as SalesOrderItemID,

      //Product
      @Search.defaultSearchElement: true
      Item.productid                as ProductID,

      //Measure fields
      Item.currencycode             as CurrencyCode,
      Item.grossamount              as GrossAmount,
      Item.netamount                as NetAmount,
      Item.taxamount                as TaxAmount,

      //BOPF Association to SO Header
      @ObjectModel.association.type: [#TO_COMPOSITION_PARENT,#TO_COMPOSITION_ROOT]
      _SalesOrder  Declare header association
}
```

Here the association between item and header imply an LEFT OUTER JOIN. If you want to force an inner join with association, example you can do something like this for example

```
      //BOPF Association to SO Header
      @ObjectModel.association.type: [#TO_COMPOSITION_PARENT,#TO_COMPOSITION_ROOT]
      _SalesOrder[1:inner].BusinessPartnerID
}
```

For more clarity on path expression joins, pls refer to here

Save and activate the ZDEMO_I_SALESORDER CDS view.

Post activation, following objects are created in Dictionary :

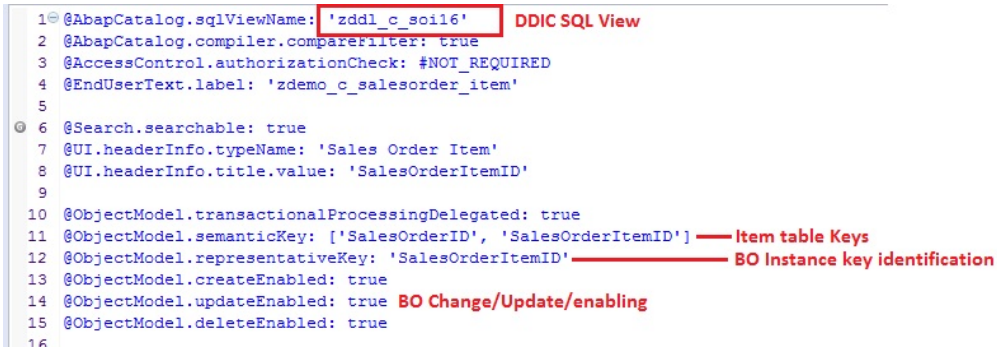
1) DB view : ZDEMO_I_SALESORDER_ITEM (CDS Entity)

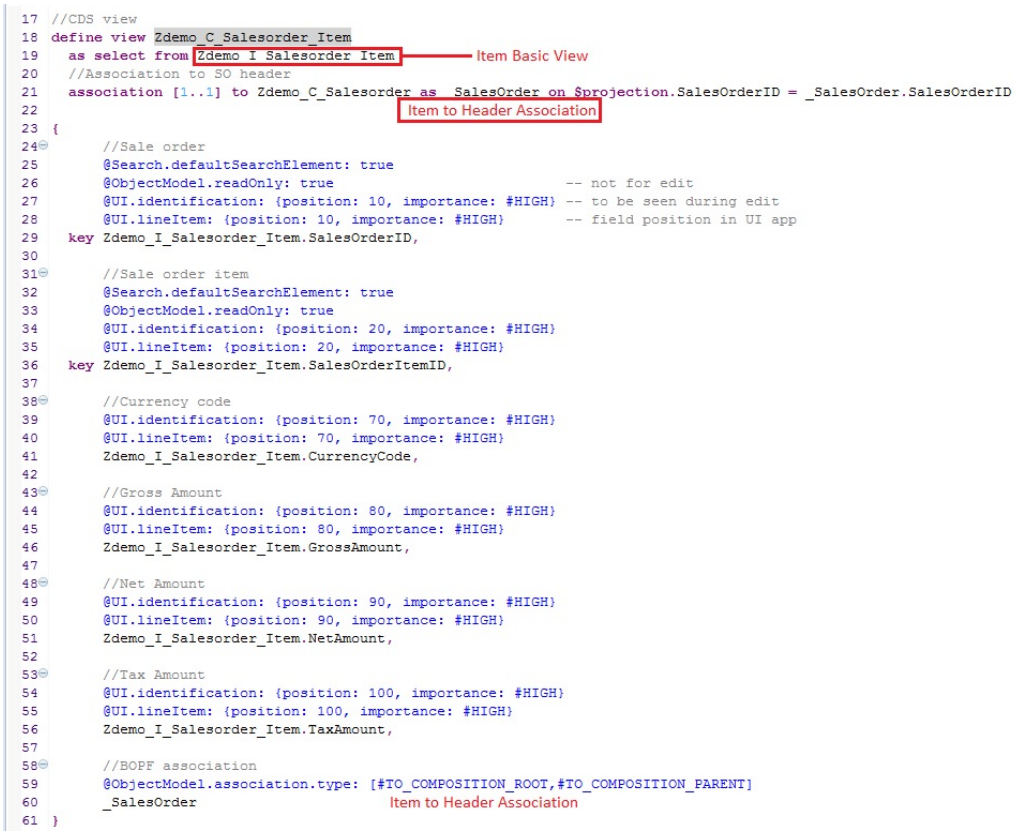
2) DD SQL view (Columnar) : ZDDL_I_SOI16

Since this Basic view cannot be used directly for UI display, an Consumption view shall be created.

Consumption SO Item CDS

The CDS code + Annotations are as below:





Save and Activate.

Post activation the objects created are in Dictionary :

1) DB view – ZDEMO_C_SALESORDER_ITEM

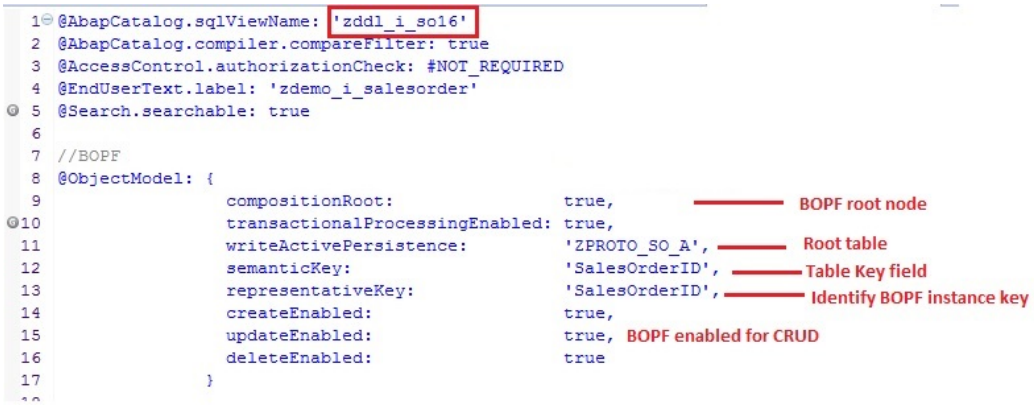
2) DD SQL View – ZDDL_C_SOI16

We are done with sales order Item.

Lets move to Sales Order Header starting with Basic view.

SO Header Basic CDS ZDEMO_I_SALESORDER

The CDS code + Annotations are:



Optional: there is an Object model annotation for modelCategory: # BusinessObject. If we use this the create sales order logic has to be written as an BOPF action. And this action shall be triggered when the + icon is clicked.


```
19 //CDS View
20 define view Zdemo_I_Salesorder
21
22   as select from zproto_so_a as SalesOrder
23
24   //Association between SO header and item
25   association [0..*] to Zdemo_I_Salesorder_Item as _Item on $projection.SalesOrderID = _Item.SalesOrderID
26
27 {
28       //Sale order
29       @Search.defaultSearchElement: true -- to be appeared as search in app
30
31   key SalesOrder.salesorderid                  as SalesOrderID,
32
33       //Business Partner
34       @Search.defaultSearchElement: true
35       SalesOrder.businesspartnerid             as BusinessPartnerID,
36
37       //Currency Code
38       SalesOrder.currencycode                  as CurrencyCode,
39
40       //Gross amount
41       SalesOrder.grossamount                   as GrossAmount,
42
43       //Net Amount
44       SalesOrder.netamount                     as NetAmount,
45
46       //Tax amount
47       SalesOrder.taxamount                     as TaxAmount,
48
49       //Lifecycle status
50       SalesOrder.lifecyclestatus               as LifecycleStatus,
51
52       //SO Item display in detail page - used for navigation
53       @ObjectModel.association.type: #TO_COMPOSITION_CHILD
54       _Item
55 }
```

Save and Activate.

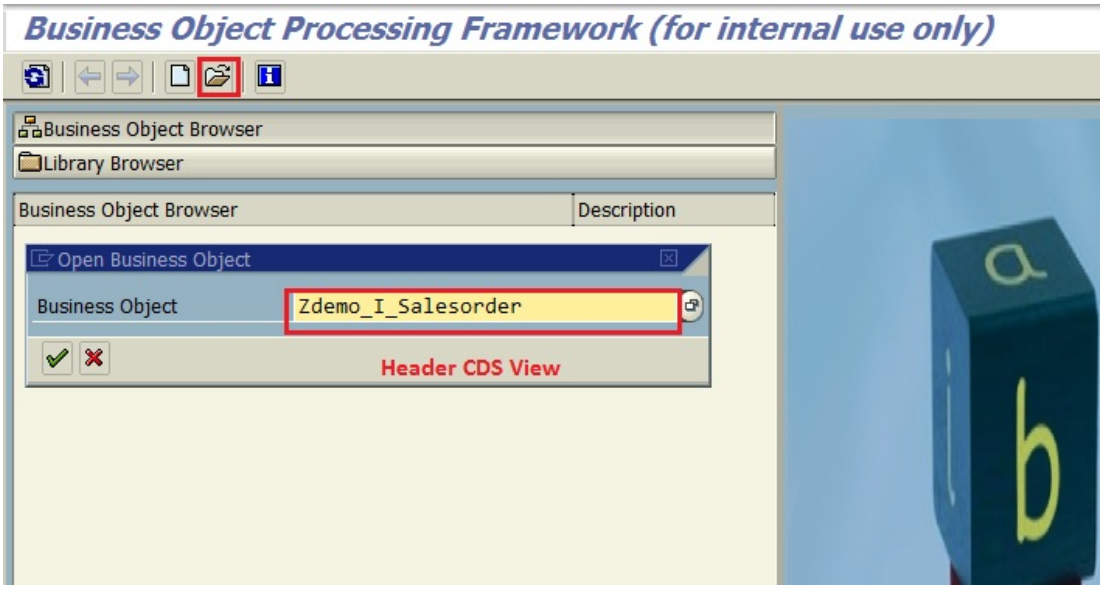
Post activation, following objects are generated ：

1) DD Sql view ZDDL_I_SO16

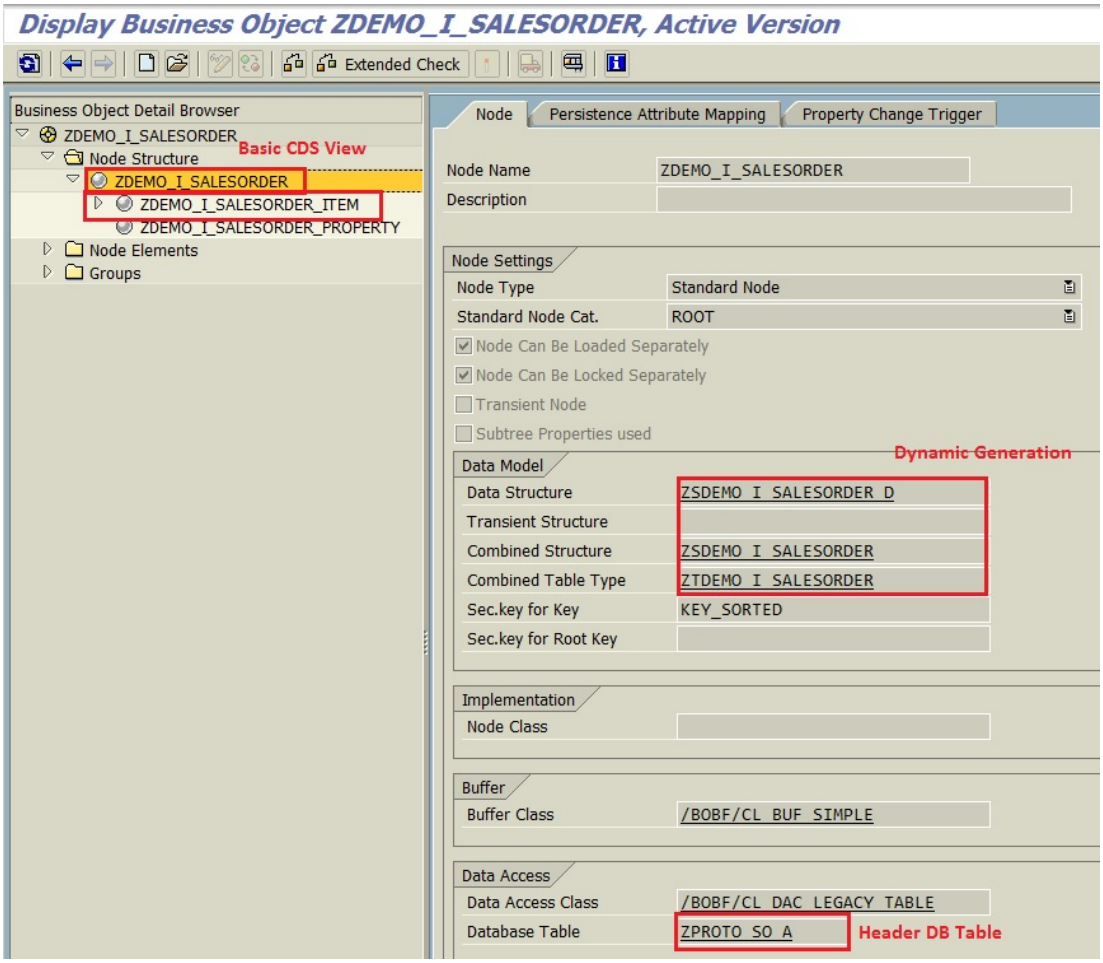
2) DB view ZDEMO_I_SALESORDER

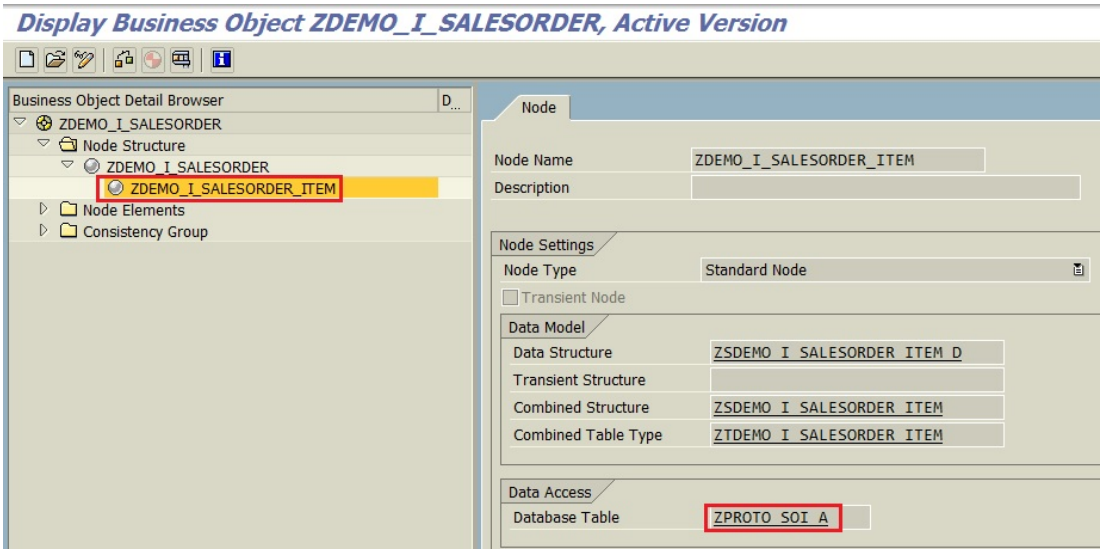
3) And an BOPF Business Object by the CDS name ie Business Object **ZDEMO_I_SALESORDER**

The BOPF Business Object can be viewed in Hana Studio/Eclipse via ABAP perspective. Go to the package, under Business Objects and here you shall see the generated BOPF object. Other option is via SAP Gui. You can open GUI in studio as well. Use Tcode BOBX or /BOBF/CONF_UI.



BOPF Business Object





Now lets use this Basic Header view in a Consumption Header View

Consumption SO Header CDS ZDEMO_C_SALESORDER

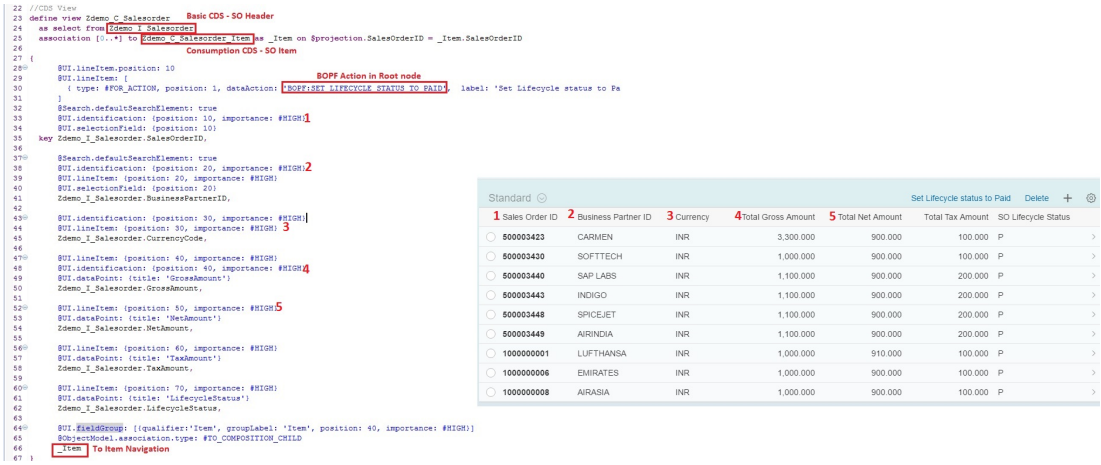
The CDS code + Annotations are:

```
1  @AbapCatalog.sqlViewName: 'zddl_c_so16'
2  @AbapCatalog.compiler.compareFilter: true
3  @AccessControl.authorizationCheck: #NOT_REQUIRED
4  @EndUserText.label: 'zdemo_c_salesorder'
5  @Search.searchable: true
6
7  //Odata
8  @OData.publish: true
9
10 //BOPF
11 @ObjectModel.transactionalProcessingDelegated: true
12 @ObjectModel.semanticKey: 'SalesOrderID'
13 @ObjectModel.representativeKey: 'SalesOrderID'
14 @ObjectModel.createEnabled: true
15 @ObjectModel.updateEnabled: true
16 @ObjectModel.deleteEnabled: true
17
18
19 @UI.headerInfo.typeName: 'Sales Order'
20 @UI.headerInfo.title.value: 'SalesOrderID'
21
```

DDIC Sql View

Publish OData

BOPF related



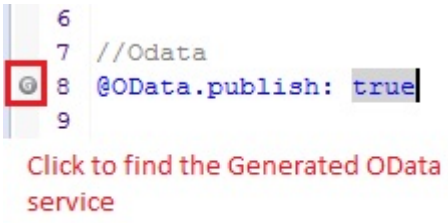
Save and Activate.

Post activation, the following objects are created:

1) DD Sql View DDL_C_SO16

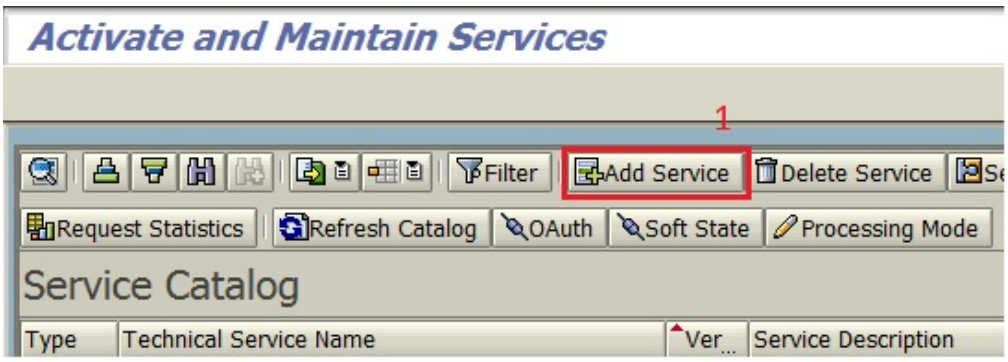
2) DB view ie CDS entity ZDEMO_C_SALESORDER (DDIC)

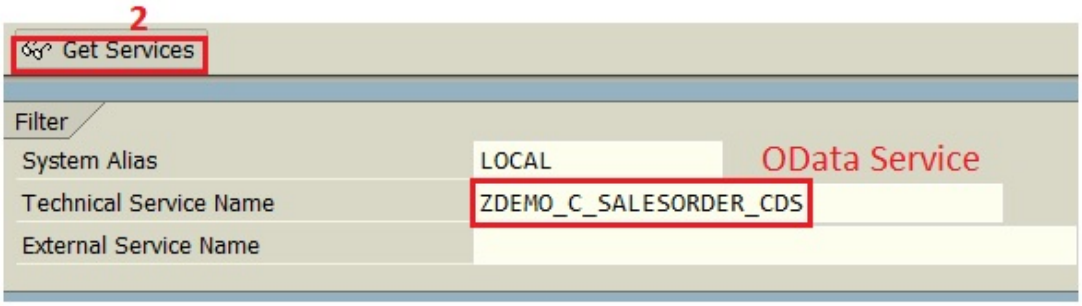
3)OData service (<CDS view>_CDS)



The OData service generated is not active. To activate go to Tcode Gui Tcode /IWFND/MAINT_SERVICE.

Click on Add service. Choose Local and enter the Name of CDS in the Technical Service and click Get services.



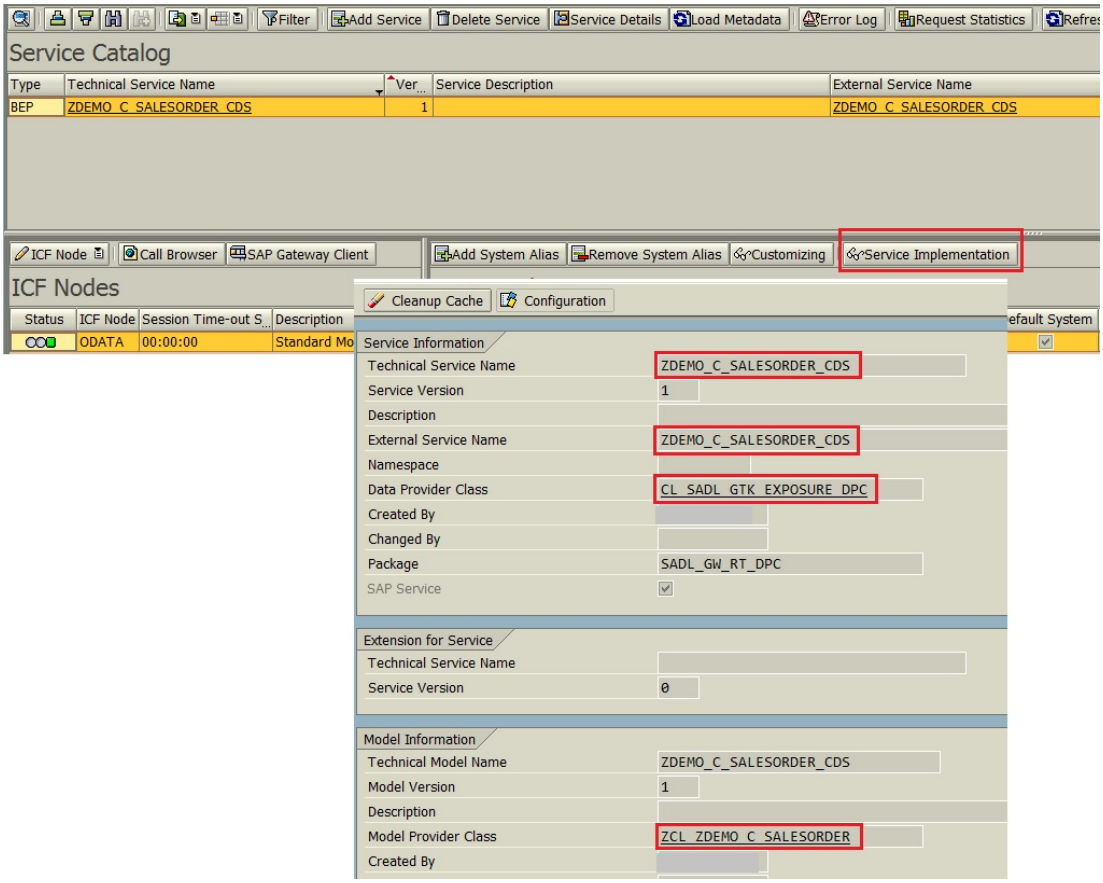


Click on “Get Services. Click on Service or click on “Add Selected Service”. In the next screen enter the package or local object and click ok. The OData service is now **Active.**

In the background when the CDS view with OData annotations get activated, the SADL generates Gateway artifacts such as Model Provider Class (MPC) and Data Provider Class (DPC) which form the backbone of OData.

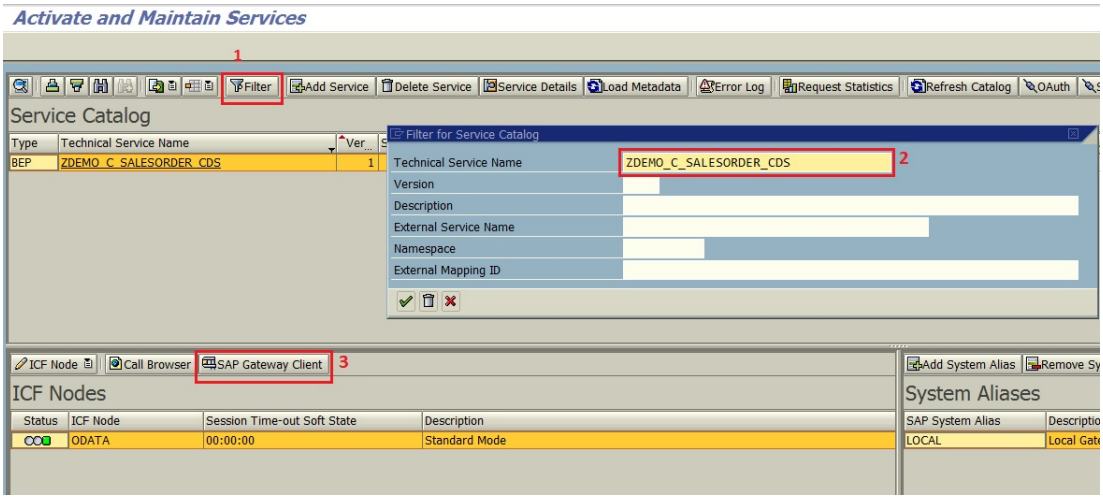
Normally to create an OData service we go to Tcode SEGW and create a project and a Data Model(Data Reference as Final Consumption CDS view) and the MPC and DPC are generated.

In our case we can view these artifacts as shown below.

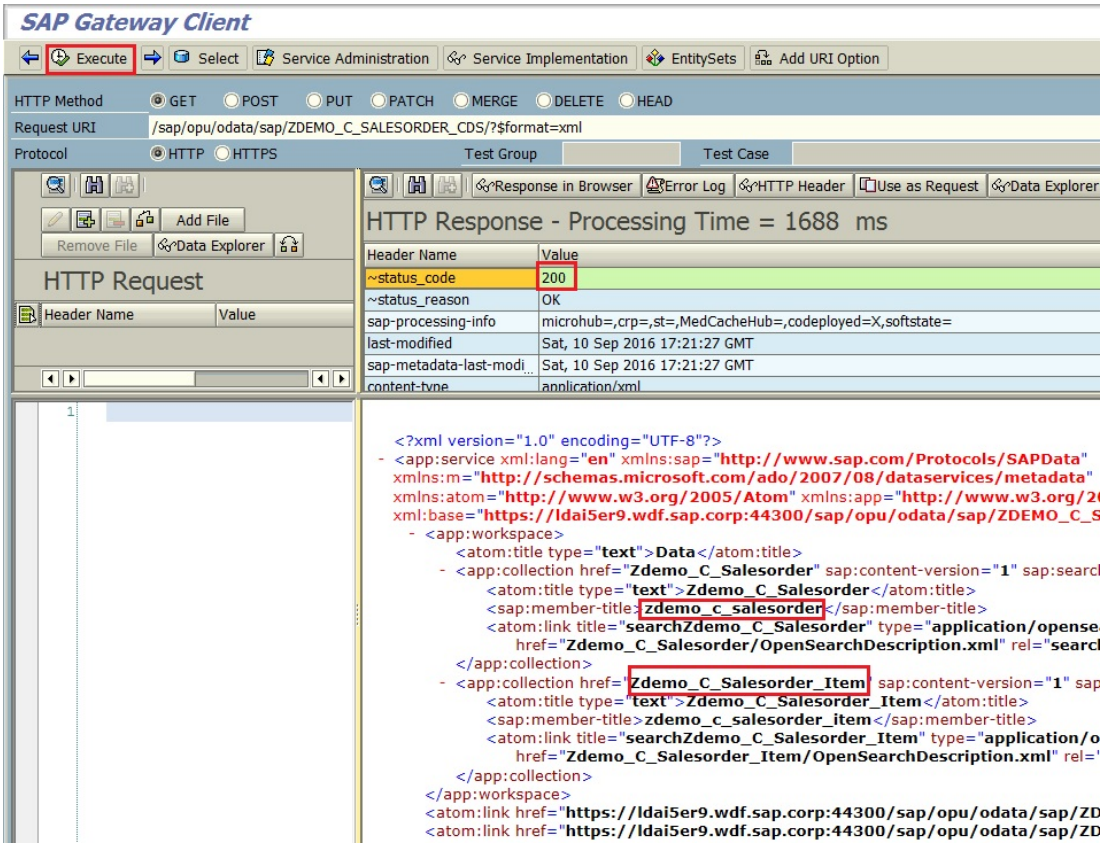


Test the Odata service.

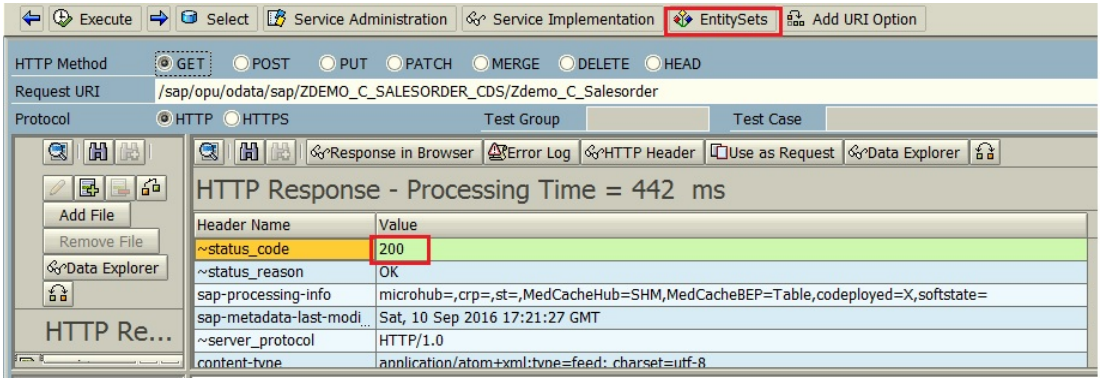
Go back to the service maintenance screen. Click on Filter and enter the service name as Technical service in our case ZDEMO_C_SALESORDER_CDS



Click on Gateway Client. In the new window click on execute. If the status is 200, all is well.



The OData metadata is fine. Test if the OData returns records by choosing the entity sets.



So we are done with Final Consumption CDS view. And we can use this view in Fiori Smart Template.

Recap of what we have done so far.

1) Created Basic Item CDS

2) Created Consumption Item CDS

3) Created Basic Header CDS

4) Created Consumption Header CDS

Before we start with Fiori App, we shall create BOPF action. The action shall change the Lifecycle Status of an Sales order

The BOPF action and the App display will be covered in the Part 2.

Alert Moderator

Assigned tags

ABAP Development

SAP Fiori

SAP Fiori for SAP ERP

SAP S/4HANA

abap cds

abap cds views

abap for hana

View more...

Similar Blog Posts

Create Fiori app using CDS with BOPF- For beginners Part 2

By Former Member Sep 20, 2016

Creating a draft enabled Sales Order Fiori App using the new ABAP Programming Model - Part 1: Overview

By Geert-Jan Klaps Mar 11, 2019

Getting Started with ABAP Core Data Services (CDS)

By Carine Tchoutouo Djomo Feb 01, 2016

Related Questions

CDS + BOPF + UI5. Call BAPI?

By Suwandi Cahyadi Jun 24, 2019

Why need multiple authorization check for FIORI app?

By Akshath LT Dec 14, 2018

Interactive / Draft Fiori Apps in NW 7.51 SP1 ?

By Attila Berencsi Feb 08, 2017

Join the Conversation

SAP TechEd

Tune in for tech talk. Stay for inspiration. Upskill your future.

Coffee Corner

Join the new Coffee Corner Discussion Group.

18 Comments

You must be Logged on to comment or reply to a post.

Former Member
October 19, 2016 at 9:14 am

hi sujin,

thanks for this tutorial. i tried to do all the steps but if i activate the basic cds view of sales order in eclipse there is no business object generated by the BOPF runtime.

Do you know why this could happen?

Like 1 | Share

Former Member | Blog Post Author
November 2, 2016 at 6:02 am

Hi Hergen,

I can think of 2 reasons why the BO must not be generated:

1) the use of @ObjectModel annotations as shown below.

```
@AbapCatalog.sqlViewName: 'zddl_i_so16'
@AbapCatalog.compiler.compareFilter: true
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'zdemo_i_salesorder'
@Search.searchable: true

//BOPF
@ObjectModel: {
                compositionRoot:                true,          ——— BOPF root node
                transactionalProcessingEnabled: true,
                writeActivePersistence:         'ZPROTO_SO_A', ——— Root table
                semanticKey:                    'SalesOrderID', ——— Table Key field
                representativeKey:              'SalesOrderID', ——— Identify BOPF instance key
                createEnabled:                  true,
                updateEnabled:                  true,  BOPF enabled for CRUD
                deleteEnabled:                  true
              }
```

2) Make sure that the DB table (in this example ZPROTO_SO_A) is available in the dictionary.

If the BO is not generated, the error messages shown during CDS activation. You can see these when you mouse over the Icon next to @objectmodel. Good luck !

Regards,
Sujin

Like 0 | Share

Jason Muzzy
March 10, 2021 at 12:55 am

I realize this question is old, but I found it while looking for a solution to the same problem. I noticed that if I activated the CDS view as a local object then it would generate the BOPF object. Based on that I found and implemented note 2948149 which allowed the BOPF object to generate even though the package was assigned to transport layer SAP.

Like 0 | Share

Jason Scott
October 20, 2016 at 3:16 am

This is a great simple example... but what has me scratching my head is that this whole CDS/BOPF thing is designed around completely separate custom data tables.

The reality of real-life is that you nearly always need to integrate with standard sap business processes.. For example you will need to call BAPI's to post purchase orders or goods movements or whatever... Not just on the write-side either - often its far easy to get the correct data out of the system by calling a bapi like BAPI_PR_GETDETAIL for example.

It seems that this technique of using CDS annotations to auto-generate a BOPF object will not work for these more real-life scenarios... Or am I missing something and it is just that all tutorials on this subject are extremely basic using a Z table?!?

Is there an "Extension" concept for these generated BOPF objects so that you can code your own ABAP update routines?

Because... if there isn't then this whole CDS/BOPF is purely academic. Great for creating mars explorer demos at a TechEd keynote... but thats about it...

Like 1 | Share

Former Member | Blog Post Author
November 2, 2016 at 5:54 am

Hi Jason,

Thank you for your comment. You are right with the real time reality scenarios. Here's my take on that.

With the new code to data Paradigm in place and with the S/4 Architecture along side, seems the CDS-BOPF combination will persists for long now.

Lets say for an write scenario, the CDS-BOPF will work just perfectly as BOPF framework takes care of the DB write/lock/buffer/validation/consistency. In a typical real time scenario, say we have some ERP tables in S4. For query purpose the CDS should work faster than a BAPI call, because the CDS is interacting with the DB layer.

Regarding extension and ABAP routines am not sure. But you can do something like this as mentioned in the Blog which is Generate the BOPF via annotations and then go to the BOPF framework and edit to add actions/determinations/validations etc.

On another note, for query/read, there are basically 2 approaches keeping in mind the code-to-data concept.
Top down approach – using CDS to interact with DB and
Bottom up approach – using HANA modelling and using the HANA artifacts for reports/query.

Regards,
Sujin

Like 1 | Share

Florian Royer
December 15, 2017 at 11:22 pm

First of all, thank you Sujin for your detailed guide !

This is exactly what I am thinking about when I hear buzzwords like BOPF & CDS in one sentence. Still cannot figure out, when to use what.

My one and only productive use of a BOPF Object was:

- Z-Table HEAD
- Z-Table ITEM

Business logic in BOPF object, vizualisation in FPM, generated by Wizards. Very simple, but yet very powerful compared to other development approaches (to be honest – I think I wrote about 200 lines of code in total, only because I wanted to implement a Drag and Drop feature, otherwise it would be less code).

I agree per 100% to Jason's post. When can I use BOPF / CDS in a real-life scenario, where I can't base on Z-Tables, like create a purchase order using custom BOPF?

The answer to this question is still open, looking forward to your replies.

Like 0 | Share

Former Member
March 13, 2017 at 5:16 am

Hi Sujin,

Thanks for the so cleared blog about cds-bopf.

Like 0 | Share

Former Member
April 21, 2017 at 1:18 pm

Hi Suijin,

thanks for the tutorial. What I need to do that I can see "Smart Template" as Project Type in SAP WebIDE? I don't get this option.

Is this already available for Partners/Csustomers?

Thomas

Like 0 | Share

Cain Sun
May 26, 2017 at 6:19 am

Thanks for the tutorial, I noticed that the DB table for SO header does not have GUID and NODE GUID fields, how can BOPF store the data? In fact, I encounter some errors during testing the result.

Like 0 | Share

Sebastian Freilinger-Huber
February 19, 2018 at 3:40 pm

Hi Cain,

as far as I know in classical BOPF you need GUIDs on the database. With BOPF in S/4 context now this has been reworked and improved by SAP.

In case for example your database table contains a common ID field or some other sort of key (instead of a GUID), you can generate the BOPF object on it as well using the corresponding CDS annotations (ObjectModel). From my understanding to make this work the ID has to be defined as key field in the CDS view, which is used for the BOPF generation.

Best regards,

Sebastian

Like 0 | Share

Joseph BERTHE
October 28, 2017 at 6:19 pm

Hello,

Thanks for your blog, I try to follow it and I encounter some difficulties 🙂 During the activation of the zdemo_i_SalesOrder CDS it through me this error :

[BO check] Element SALESORDERITEMID is not a DB field (view ZDEMO_I_SALESORDER_ITEM, table ZPROTO_SOI_A)

As you can imagine, the field **SALESORDERITEMID** is in the table.

What could be the problem ?

Regards

Like 1 | Share

Former Member
November 18, 2017 at 8:21 pm

Hello,

Thanks for the blog.

I would like to try the tutorial, but i don't know where i get the tables mentioned in the tutorial. So far i only know sflight.

Regards

Dennis

Like 0 | Share

Sourabh Gandhi
November 30, 2017 at 7:51 am

Thanks Sujin. This was very helpful. Please provide more complex scenarios. Regards Prateek Sonthalia

Like 0 | Share

Sebastian Freilinger-Huber
February 19, 2018 at 3:46 pm

Hi Sujin,

thanks for this blog. Just one question considering the maintenance of the BO Objects.

Is there a reason why you don't use Eclipse as you are already working on a 7.50 system?

Best regards,

Sebastian

Like 0 | Share

Terry Huang
March 9, 2018 at 1:57 am

Hi,

why we need to create seprate cusumption and basic views, I just tried only create two basic views. it wroks properly.

Best regards,

Terry

Like 0 | Share

Former Member
April 25, 2018 at 9:59 pm

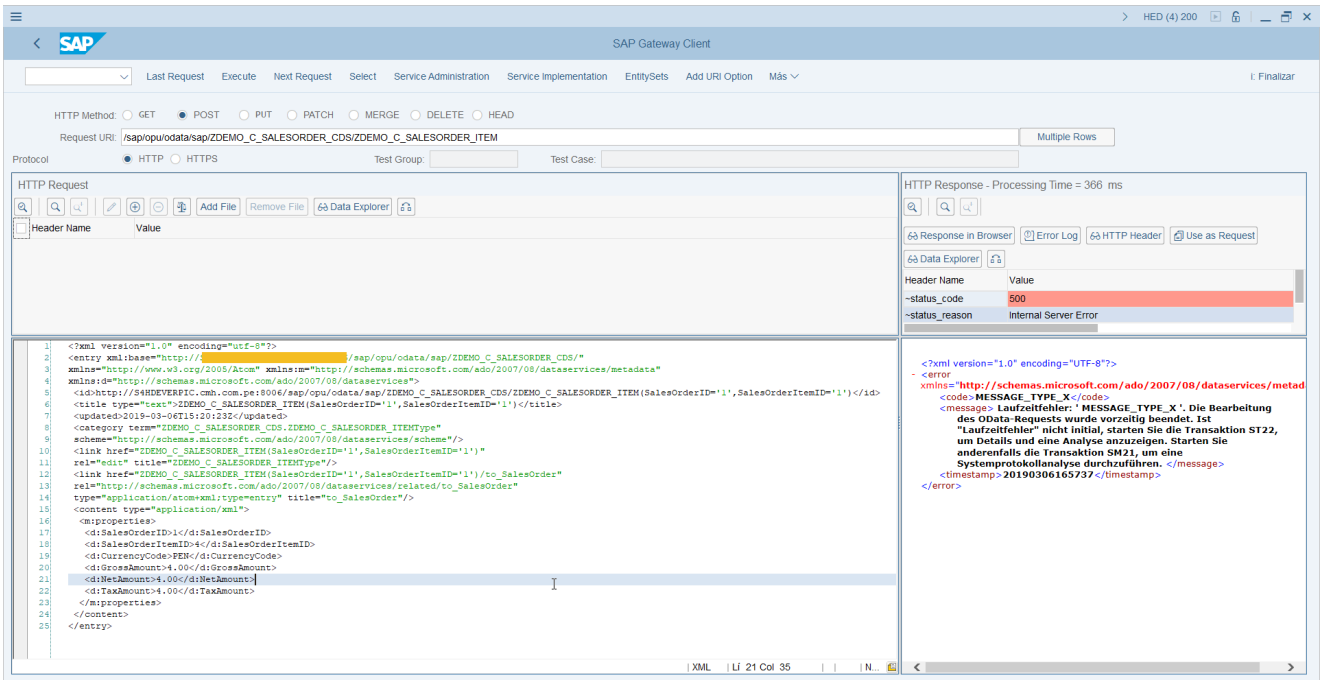
I tried creating 2 basic views but if I want to expose to UI then what should be done in this situation. How is consumption view different from basic view.

Need suggestions

Like 0 | Share

IDMIR LEVI DE LA CRUZ CLEMENTE
March 6, 2019 at 10:12 pm

Hi sujin Thanks for this blog.
I'm trying to consume the service to insert the detail and I'm getting the follow



when I register the cabezera I have no problem.

The CDS is the following

```
@AbapCatalog.sqlViewName: 'ZDDL_I_SO16'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'ZDEMO_I_SALESORDER'
@Search.searchable: true

@ObjectModel:{
    compositionRoot: true,
    transactionalProcessingEnabled: true,
    writeActivePersistence: 'ZPROTO_SO_A',
    semanticKey: ['SalesOrderID'],
    representativeKey: 'SalesOrderID',
    createEnabled: true,
    updateEnabled: true,
    deleteEnabled: true
}

define view ZDEMO_I_SALESORDER
  as select from zproto_so_a as SalesOrder
  association [0..*] to ZDEMO_I_SALESORDER_ITEM as _Item on $projection.SalesOrder

{

      @Search.defaultSearchElement: true

  key SalesOrder.salesorderid       as SalesOrderID,

      @Search.defaultSearchElement: true

      SalesOrder.businesspartnerid  as BusinessPartnerID,

      SalesOrder.currencycode       as CurrencyCode,

      SalesOrder.grossamount        as GrossAmount,
```

```
      SalesOrder.netamount          as NetAmount,

      SalesOrder.taxamount          as TaxaMount,

      SalesOrder.lifecyclestatus    as LifecycleStutus,

      @ObjectModel.association.type: #TO_COMPOSITION_CHILD

      _Item

}
```

Like 0 | Share

Kai Sicker
June 28, 2020 at 10:53 pm

Thanks a lot!

Like 0 | Share