Create an ABAP Core Data Services (CDS) View in SAP Cloud Platform, ABAP Environment

Julie Plummer January 18, 2021

Created by  July 9, 2020

Beginner 45 ABAP Development, Tutorial,

Series that include this tutorial:

GROUP: Create and Enhance a CDS View in SAP Cloud Platform, ABAP Environment

min. SAP BTP, ABAP environment, Beginner

Create a CDS View, display it in Fiori Elements preview, and enhance its appearances using built-in annotations

You will learn

- How to create a CDS view with parameters
- How to display your CDS view in a Fiori Elements preview
- How to add selection fields to Fiori Elements preview
- How to extract the metadata of your CDS view
- How to add semantic annotations
- How to add a search function
- How to add selection fields

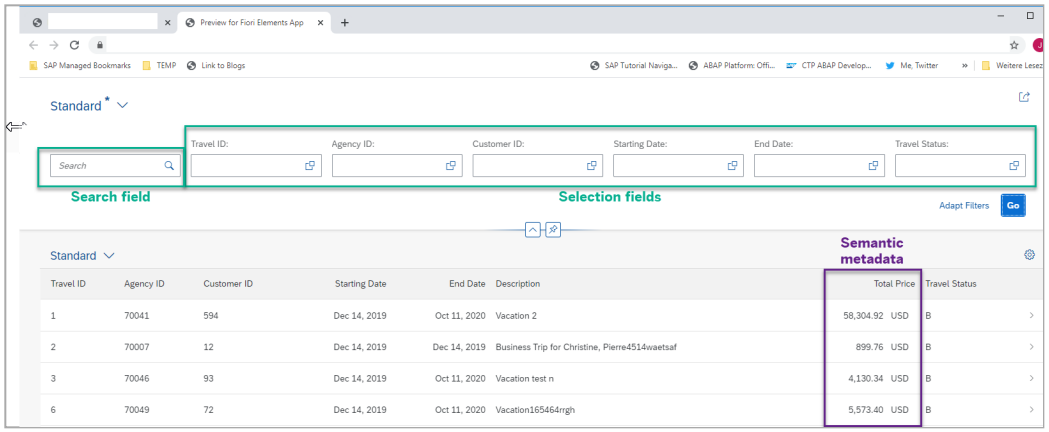
Provide Feedback

Provide Feedback

Prerequisites

- You have done one of the following:
  - **Tutorial**: Create an SAP Cloud Platform ABAP Environment Trial User
  - You have a licensed version of SAP Cloud Platform, ABAP Environment
- You have installed ABAP Development Tools, latest version
- You have downloaded the ABAP Flight Reference Scenario. To pull this reference scenario from `Github`, see Downloading the ABAP Flight Reference Scenario

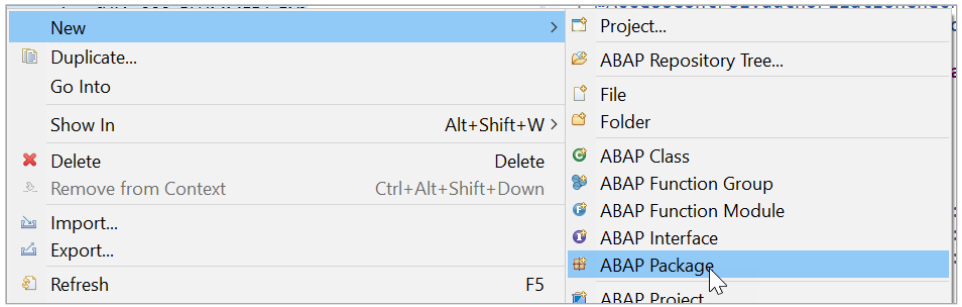
You can then use some of these features in productive development to make your applications more powerful and more user-friendly. By the end of this tutorial, your application should look like this.



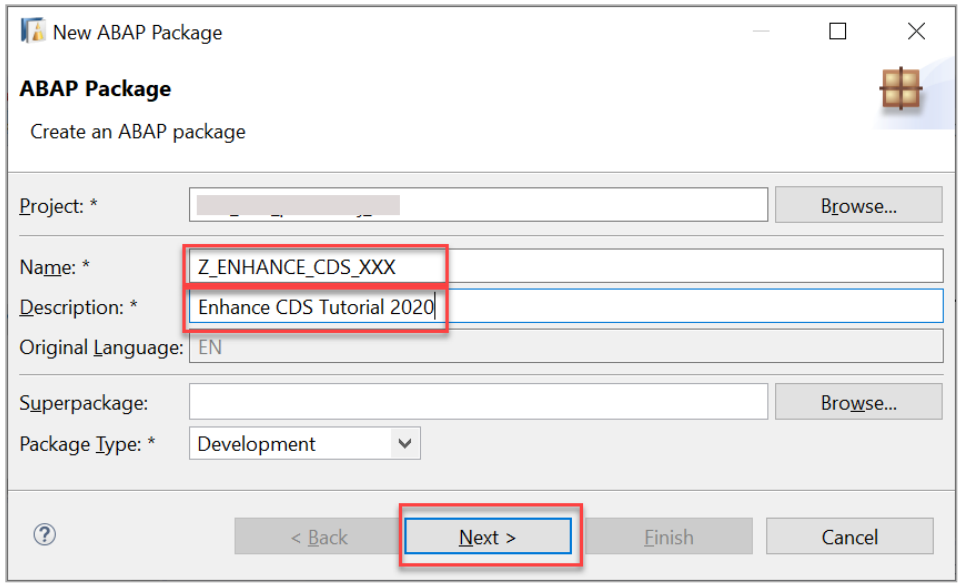
Throughout this tutorial, object names may include a suffix or group number, such as `xxx`. Always replace this with your own group number or initials.

1 Step 1: Create package

1. Create a new package for this tutorial, by choosing **New > ABAP Package**.



2. Enter a name `Package Z_ENHANCE_CDS_XXX` and description **Enhance CDS Tutorial 2020**, then follow the wizard.

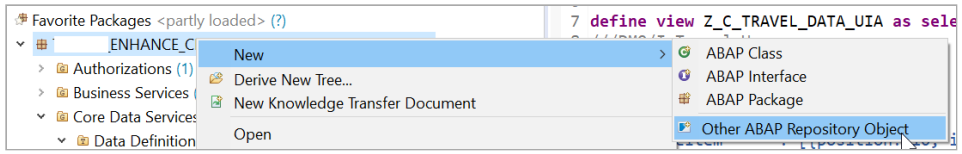


Done

Log on to answer question

Step 2: Create CDS View

1. In your package, create a CDS view. Select the package, then choose **New > Other** from the context menu, then choose **Data Definition**.



2. Add a name, `Z_C_TRAVEL_DATA_XXX`, and description `Consumption view from /DMO/I_TRAVEL_U`.

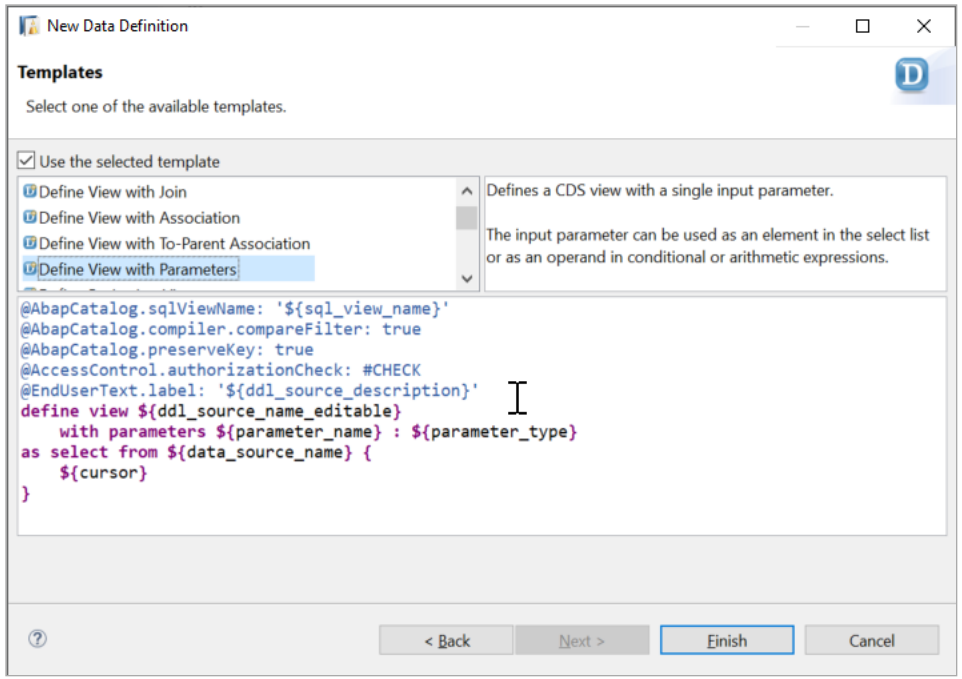
   Your CDS view is a consumption view. It is based on the business object (BO) view, `/DMO/I_TRAVEL_U`. which view provides a given data model independent of the consumption layer. It contains all core information required by applications running on top of it.

   A consumption view is a CDS view defined on top of a BO view and is used:

   - to expose the fields fitting to a given consumption use case
   - to enrich the data model with metadata using annotations (e.g. for UI, search, and OData)

3. Choose or create a transport request, then choose **Next**. Do not choose **Finish.**

4. Finally, choose **Use template** then choose **Define view**. Then choose **Finish**.



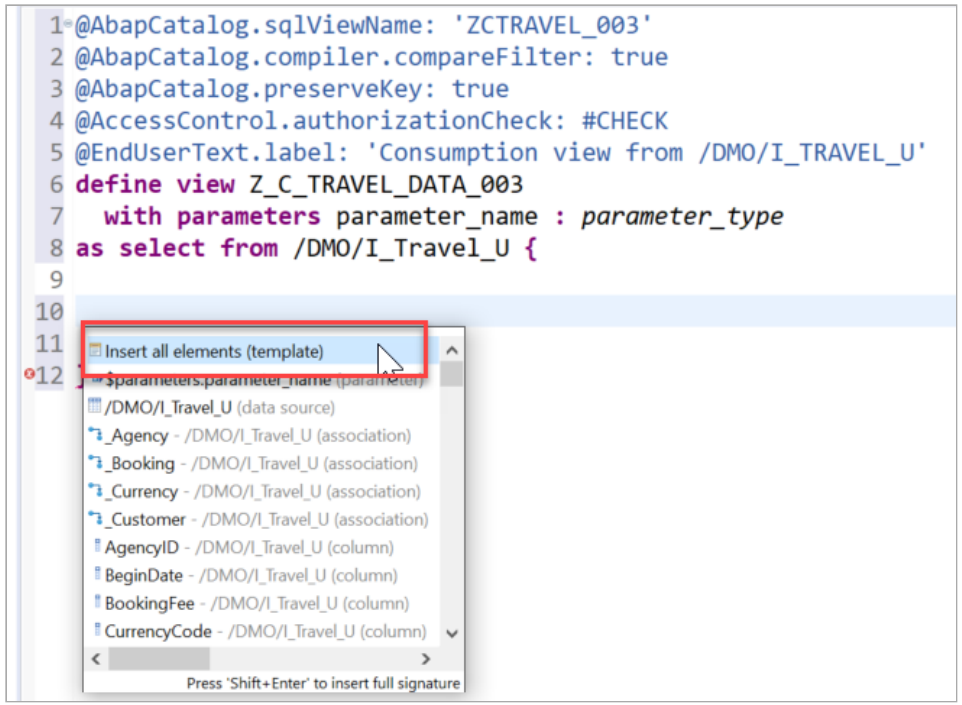
Your CDS view appears in a new editor.

Done

Log on to answer question

3 Step 3: Define CDS View

1. Add the following:
   - `sql_view_name` = `ZCTRAVEL_XXX`
   - `data_source_name` = `/DMO/I_Travel_U`. You can use **Auto-Complete** `Ctrl+Space`
2. Insert all the elements from `/DMO/I_TRAVEL_U` by placing your cursor inside the `as select from` statement (curly brackets) and again choosing **Auto-Complete** `Ctrl+Space` .



3. Comment out the statement `with parameters parameter_name : parameter_type` for now, so that the error disappears.

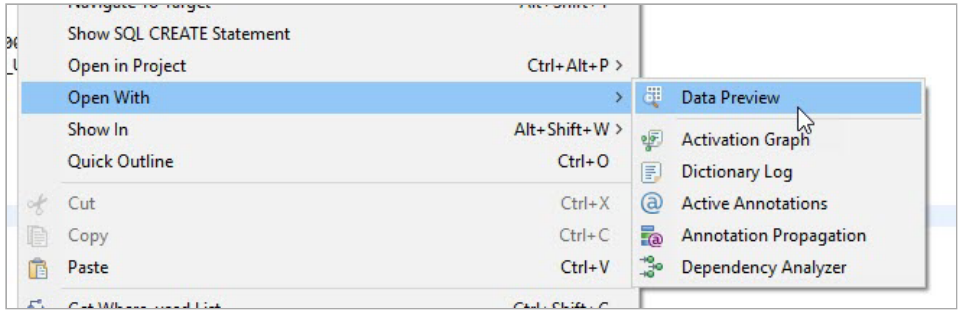
4. Format, save, and activate your code by choosing `Shift+F1`, `Ctrl+S, Ctrl+3`.

Done

Log on to answer question

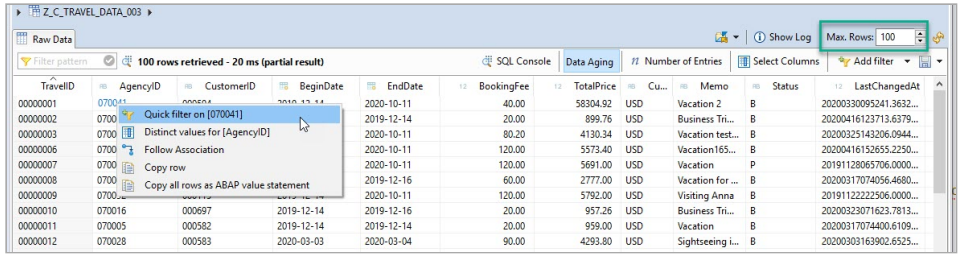
4 Step 4: Display in Data Preview

1. Click anywhere in the editor and choose **Open With > Data Preview** from the context menu.



2. The Data Preview is displayed in a new tab. You can investigate the data, by filtering, specifying the number of rows, and so on.

The values in `LastChangedAt` are not user-friendly, but you solve that by providing a Fiori elements preview in the next step.



Done

Log on to answer question

5 Step 5: Create a service definition

You will now expose the CDS view as a **business service**. This will allow you to preview your changes in Fiori elements preview.

A **business service** consists of a **service definition** and a **service binding**.

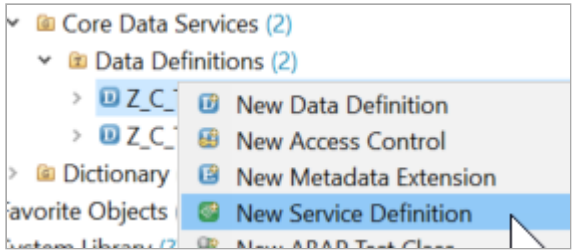
You use a **service definition** to define which data is to be exposed (with the required granularity) as a Business Service.

You then use the **service binding** to bind a service definition to a client-server communication protocol such as OData. This allows you to provide several bindings for the same definition, e.g. to expose the service to a UI, and to an `A2X` provider.

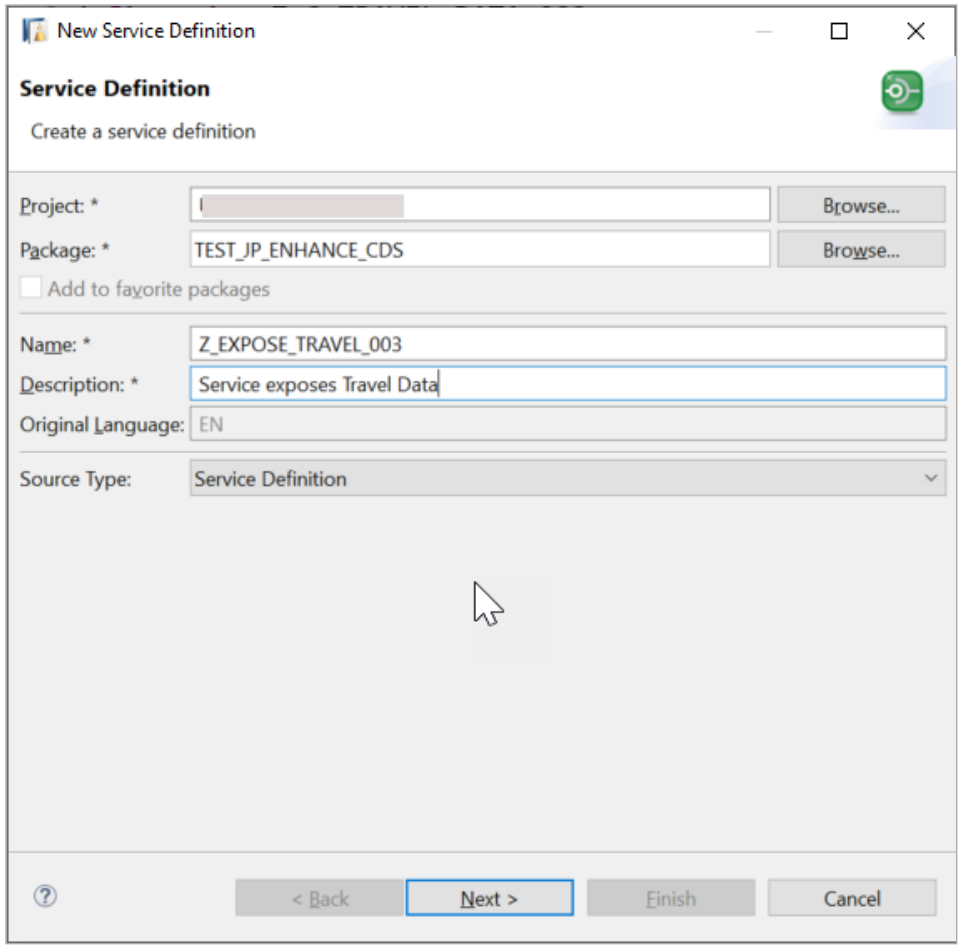
For more information, see:

- SAP Help Portal: Creating a Service Definition
- SAP Help Portal: Creating a Service Binding

1. First, create the service definition, by selecting your CDS view and choosing **New > Service Definition** from the context menu.

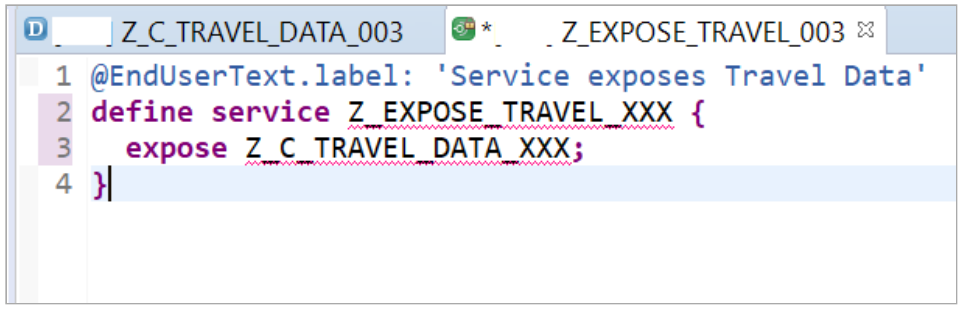


2. Choose a name and description:
    - `Z_EXPOSE_TRAVEL_XXX`
    - **Service exposes Travel data**



3. Choose the transport request; choose **Next**.

4. Use the selected template; choose **Finish**. The name of your custom entity is inserted automatically.



5. To make the service definition more readable, add an alias to the **expose** statement:

CDS Copy

```
expose Z_C_TRAVELS_XXX as Travel;

```

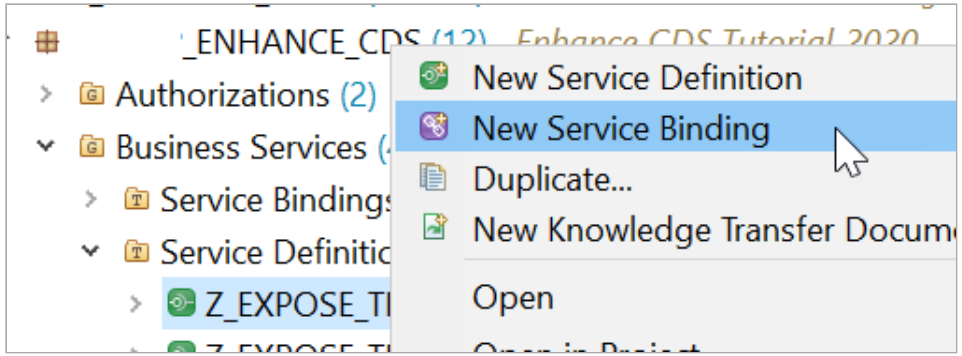
6. Format, save, and activate (`Shift+F1, Ctrl+S, Ctrl+F3`) the service definition.

Done

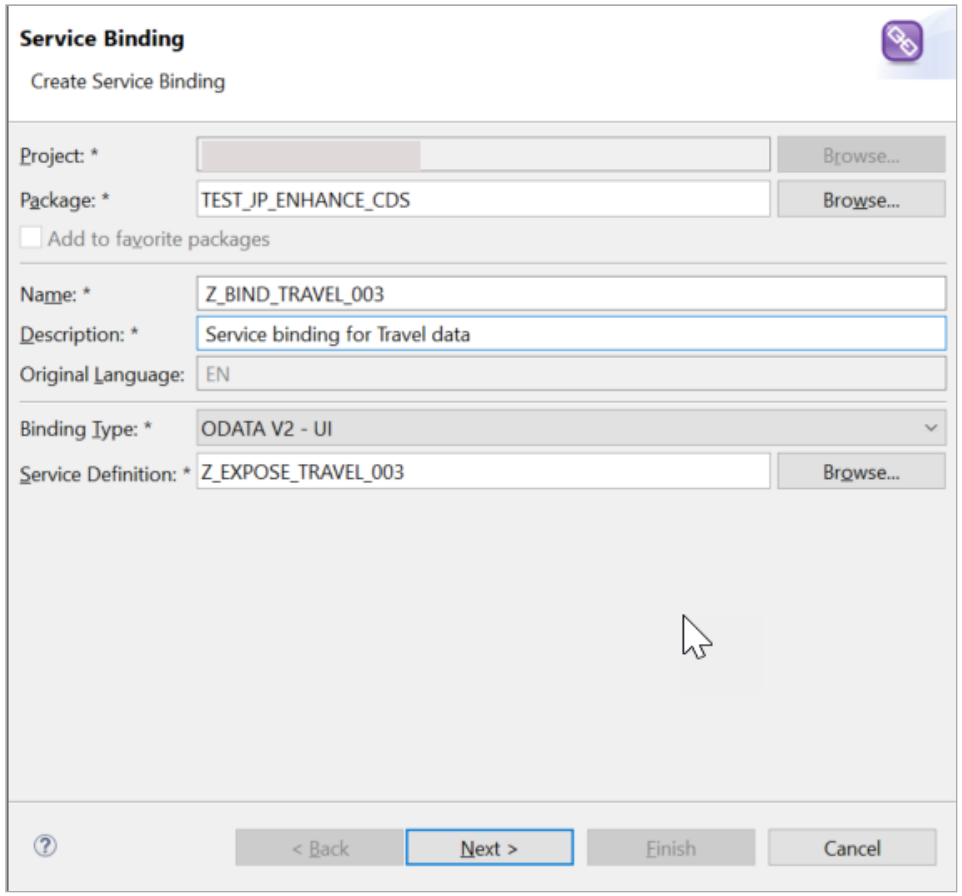
Log on to answer question

6 Step 6: Create service binding

1. Select your service definition, then choose **Service Binding** from the context menu, then choose **Next**.



2. Choose:
    - Name = `Z_BIND_TRAVEL_XXX`
    - Description = **Service binding for Travel data**
    - Binding Type = **ODATA V2 (UI...)**
    - Service Definition = `Z_EXPOSE_TRAVEL_XXX`



3. Choose the transport request; choose **Finish**.

The service binding automatically references the service definition and thus the exposed custom entity.

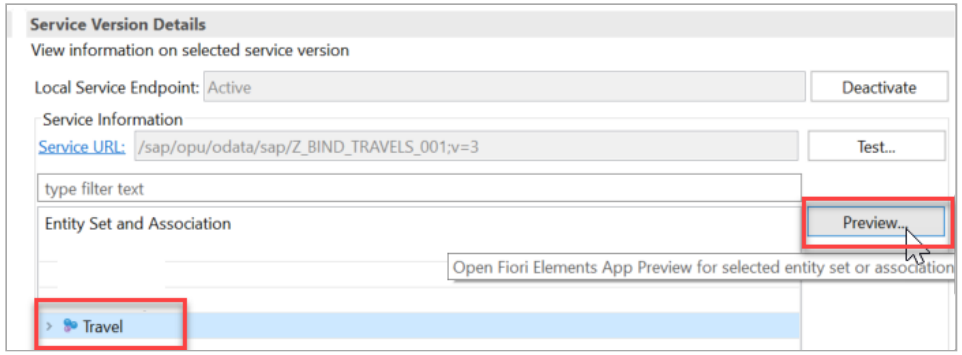
Done

Log on to answer question

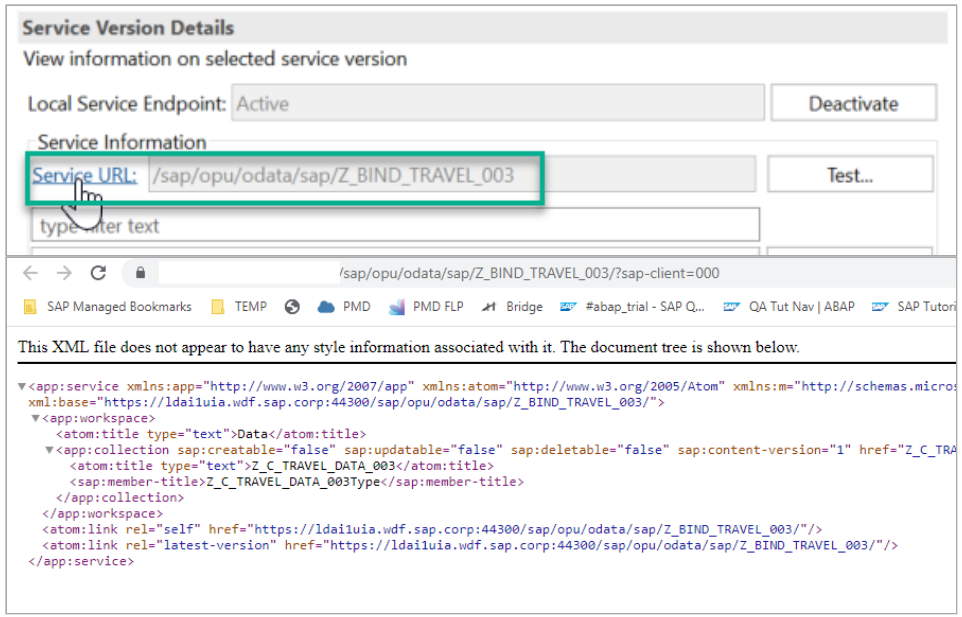
7 Step 7: Activate service binding

1. In the editor that appears, choose **Activate**.

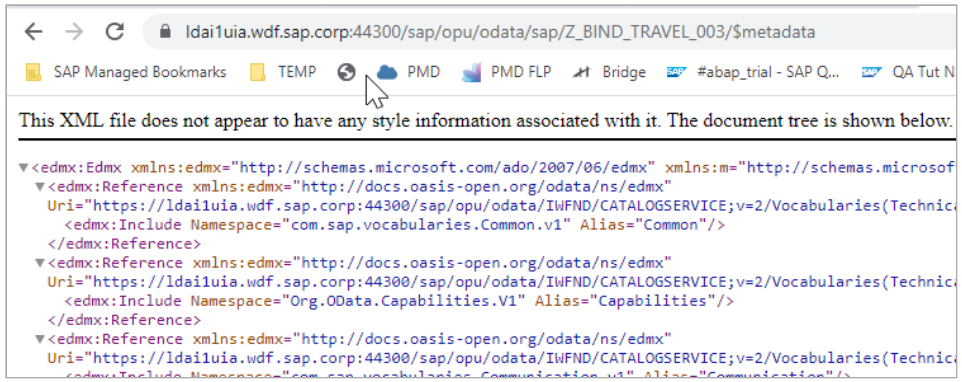
2. You can now see the Service URL and Entity Set.



3. To open the Service Document (`XML`) in your browser, choose **Service URL**.



4. In the browser, you can also see the **Metadata Document** of the Business Service by adding $metadata to the URL:
`/sap/opu/odata/sap/Z_BIND_TRAVEL_XXX/$metadata` .

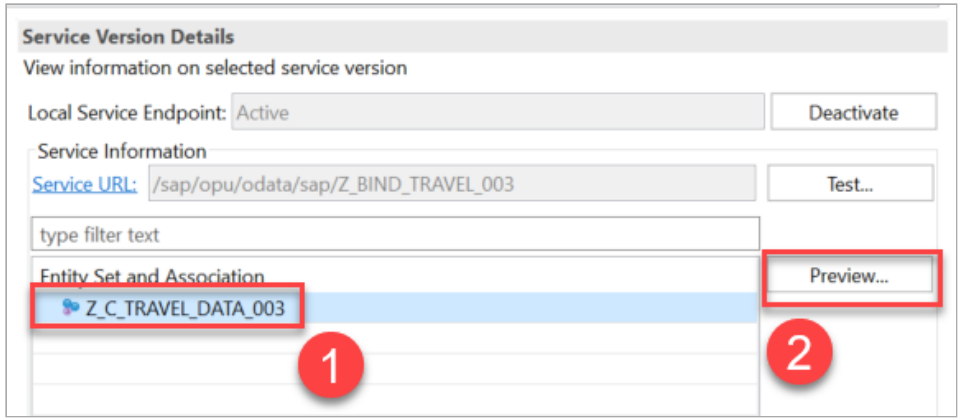


Done

Log on to answer question

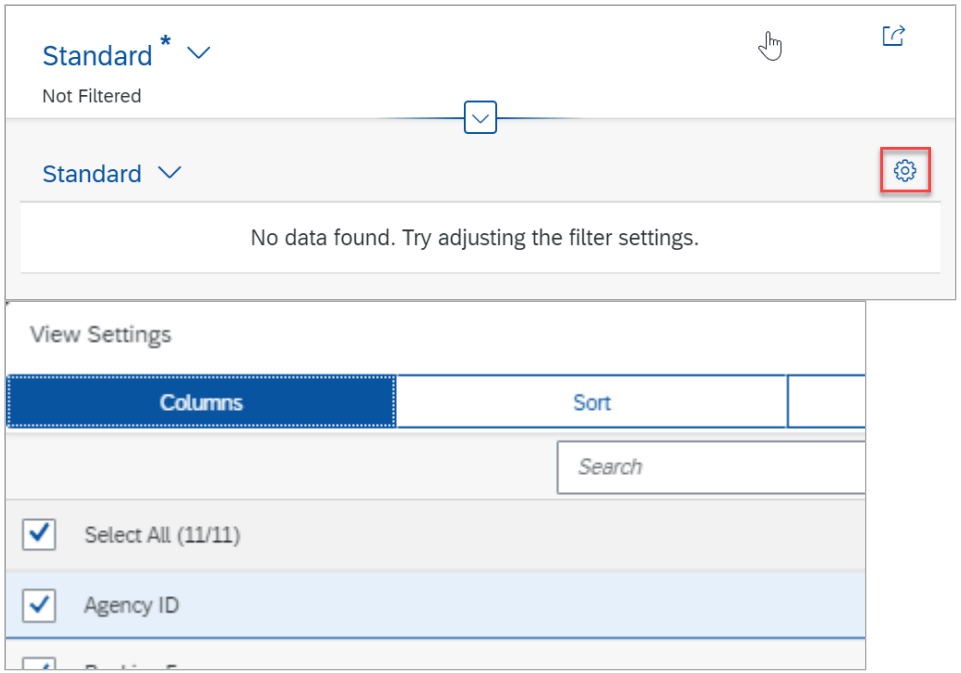
8 Step 8: Display Fiori Elements Preview

1. Select the entity set and choose **Preview**.

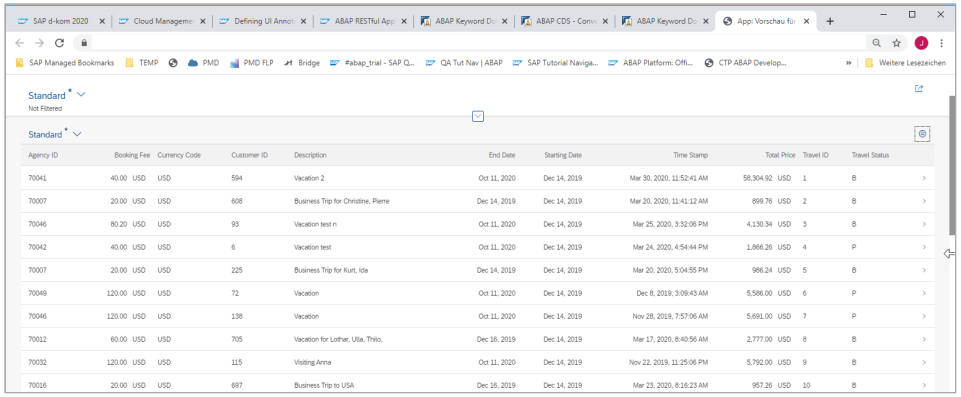


2. Log in using your ABAP Environment user and password; the Fiori Elements preview appears.

3. By default, no columns are selected. To see the data, choose **Settings**, then choose **Select All**.



4. Display the data by choosing **Go**.



Done

Log on to answer question

9 Step 9: Add annotation for automatic display

1. It would be nice if at least some fields were displayed immediately for the user. To do this, simply add the following annotation to the relevant fields in `Z_C_TRAVEL_DATA_XXX`. The start of the CDS view will then look like this.

   > `BookingFee` is not automatically displayed. The numbers for each field are relative to the other fields and are responsive - they do not refer to a specific pixel position or similar. For larger entities, you can specify *HIGH*,*MEDIUM*, or *LOW*, so that less important fields are automatically hidden on a smaller screen, such as a mobile phone.

CDS Copy

```
@UI            : {
lineItem       : [{position: 10, importance: #HIGH}]
}
@UI            : {
lineItem       : [{position: 10, importance: #HIGH}]
}
key TravelID;

@UI            : {
      lineItem       : [{position: 15, importance: #HIG
```

```
11        }
12  AgencyID,
13
14  @UI           : {
15        lineItem      : [{position: 20, importance: #HIG
16        }
17  CustomerID,
18
19  @UI           : {
20        lineItem      : [{position: 30, importance: #HIG
21        }
22  BeginDate,
23
24  @UI           : {
25        lineItem      : [{position: 40, importance: #HIG
26        }
27  EndDate,
28
29  BookingFee,
30
31  @UI           : {
32        lineItem      : [{position: 50, importance: #HIG
33        }
34  TotalPrice,
35
```

2. If you now refresh your Fiori Elements preview, you will notice that you do not have to choose the fields; you simply have to choose **Go**.

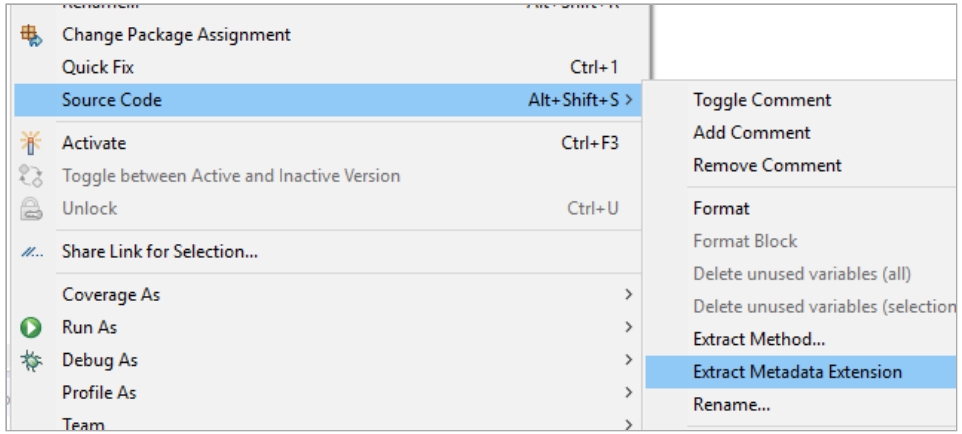
Done

10 Step 10: Extract UI metadata

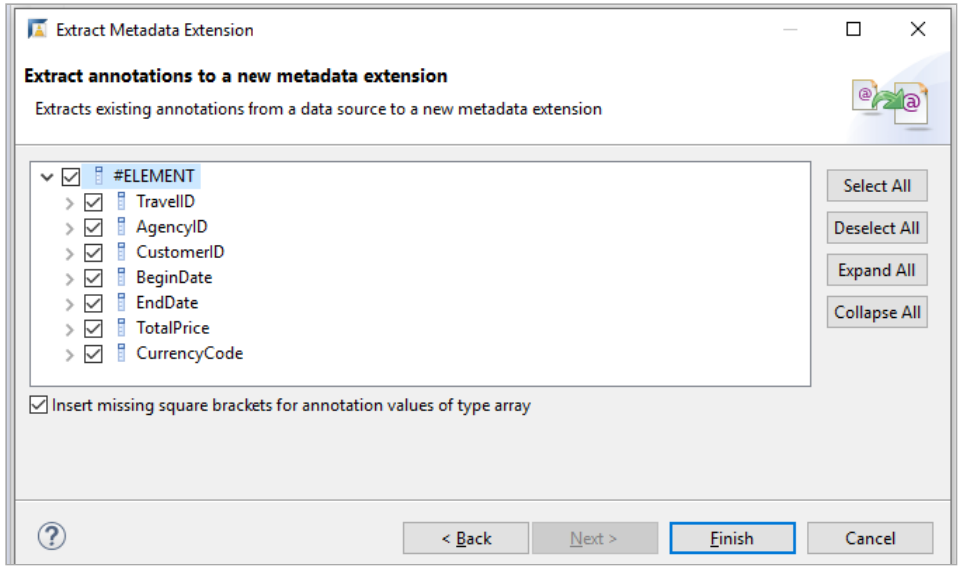
At present, you only have minimal annotations. As you add more, your CDS view will start to get cluttered. So you should extract your UI annotations to a separate object, a **metadata extensions** object, as follows:

1. First add the annotation `@Metadata.allowExtensions: true` to your CDS view.

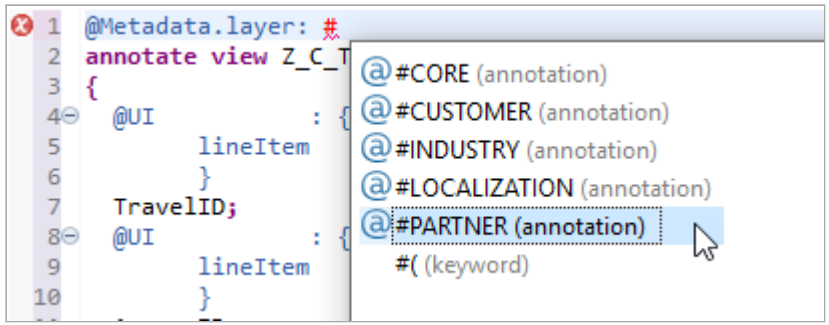
2. Then, click anywhere in the editor, then choose **Source Code > Extract Metadata Extension** from the context menu.



3. Enter a name and description for your metadata extension object, clearly similar to your CDS view name, and choose **Next**:
    - `Z_TRAVEL_METADATA_XXX`
    - `Metadata for Z_C_TRAVEL_DATA_XXX`
4. Accept the transport request, choose **Next**, select all elements, then choose **Finish**.



5. You will get an error, because you have not yet assigned the metadata extension to a layer. Since you are in sandbox mode, enter the value `#CORE` using auto-complete (**Ctrl+Space**).



> Layers allow customers or partners, for example, to enhance the metadata without modifying the CDS entity. You can also add industry- or country-specific enhancements.
>
> The metadata extensions are evaluated in a specific order. For more information, see Annotation Propagation.

6. Format, save, and activate (`Shift+F1, Ctrl+S, Ctrl+3`).

Done

Log on to answer question

11 Step 11: Add semantic metadata

If you define currency amounts and currency codes semantically, then the system will apply specific rules to handle these fields appropriately.For example, in this tutorial, if you define `TotalPrice` as a currency amount, and define `CurrencyCode` as a currency code field, then the system will add the appropriate currency to the `TotalPrice` column automatically. There is no need to display `CurrencyCode` as a separate column.

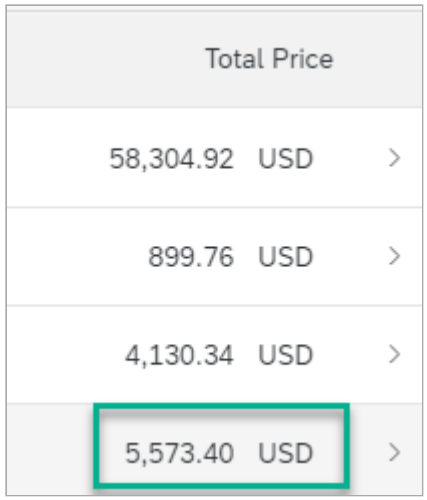
1. To do this, add the following two annotations to your CDS view:

CDS Copy

```
@Semantics.amount.currencyCode: 'CurrencyCode'
TotalPrice,

@Semantics.currencyCode
CurrencyCode,

```

2. Format, save, and activate (`Shift+F1, Ctrl+S, Ctrl+3`).

3. If you refresh the Fiori Elements preview, the **Total Price** column now looks like this.



Done

Log on to answer question

12 Step 12: Add search field

You will now add a fuzzy search capability.

1. First, add the search annotation to your CDS view:

CDS Copy

```
@Search.searchable: true
```

2. Then add the following two annotations to the field you want to search, in this case `Memo`:

CDS Copy

```
@Search.defaultSearchElement: true
@Search.fuzzinessThreshold: 0.90

```

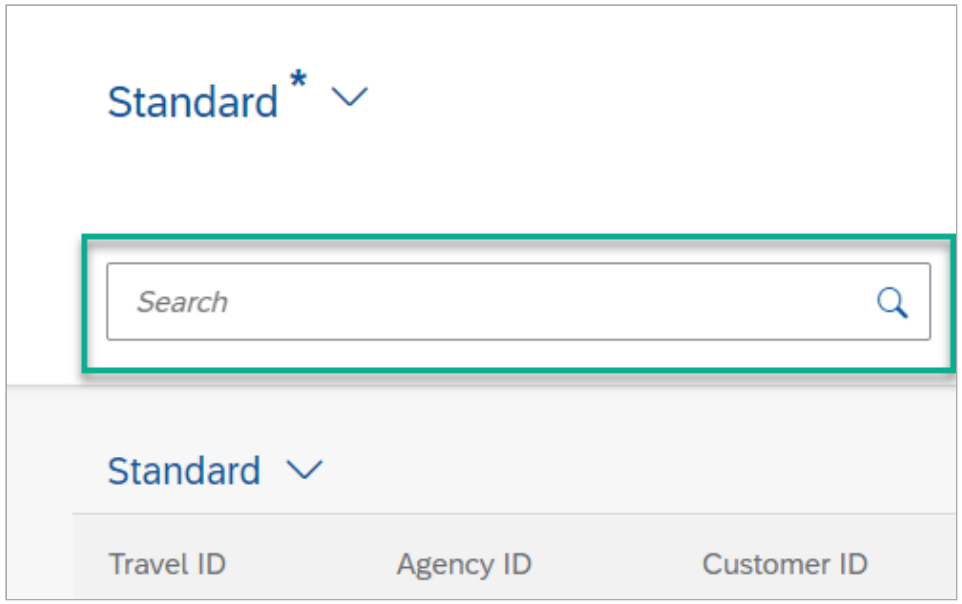
3. For convenience, add the following annotation to the metadata extension object, so that the **Memo** field appears by default in the preview, then format, save, and activate (`Shift+F1, Ctrl+S, Ctrl+3`):

CDS Copy

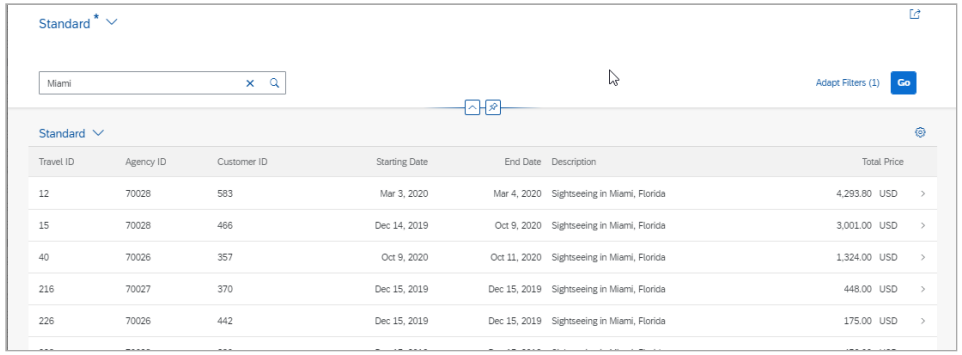
```
@UI            : {
       lineItem       : [{position: 60, importance: #HIG
       }
Memo;

```

4. Refresh the Fiori elements preview in your browser.

5. There is a new **Search** input field.



6. Enter the search text **Miami**. The app only displays trips to Miami (to date, eleven trips).



7. Optional: You can test the fuzziness threshold by changing the value to 0.70. After you save and activate, the app will now show trips to Miami and trips involving Matthias. (You may need to empty the cache.)

226	70026	442	Dec 15, 2019	Dec 15, 2019	Sightseeing in Miami, Florida
228	70028	380	Dec 15, 2019	Dec 15, 2019	Sightseeing in Miami, Florida
230	70026	455	Dec 15, 2019	Dec 15, 2019	Sightseeing in Miami, Florida
233	70027	288	Dec 15, 2019	Dec 15, 2019	Sightseeing in Miami, Florida
234	70028	268	Dec 15, 2019	Dec 15, 2019	Sightseeing in Miami, Florida
275	70029	486	Dec 15, 2019	Dec 15, 2019	Sightseeing in Miami, Florida
528	70030	610	Oct 10, 2020	Oct 10, 2020	Sightseeing in Miami, Florida
1186	70010	365	Dec 13, 2019	Dec 13, 2019	Business Trip for Juan, Amelie, Matthias
1345	70006	90	Dec 14, 2019	Dec 14, 2019	Business Trip for Adam, Matthias, Jean-Luc
1356	70006	660	Dec 14, 2019	Dec 14, 2019	Business Trip for James, Matthias

Done

Log on to answer question

13 Step 13: Add selection fields

As well as search fields, you can filter the list using an input field. In the next tutorial, you will provide input value help for these fields.

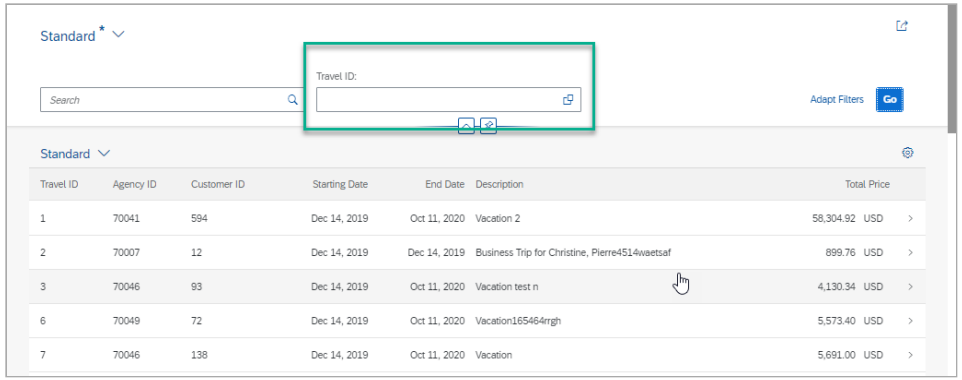
1. Add the `selectionField` annotation to the field `TravelID` in your metadata extension file, so that the whole UI annotation looks like this:

CDS Copy

```
@UI           : {
      lineItem      : [{position: 10, importance: #HIG
      selectionField: [{position: 10 }]
```

```
4         }
5 TravelID;
```

2. Format, save, and activate (`Shift+F1, Ctrl+S, Ctrl+3`). The Fiori elements preview should now look like this:



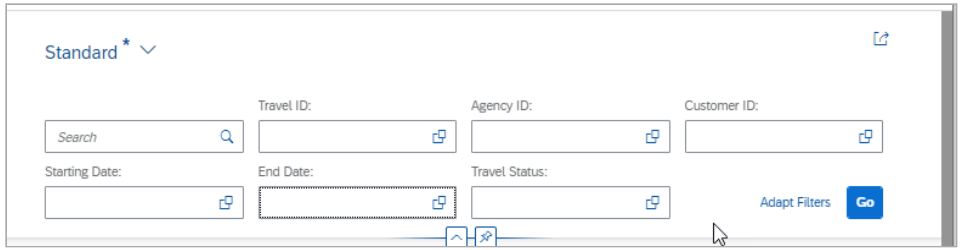
3. Add other fields as input fields by adding the following to the metadata extensions file, so that the file looks like this:

CDS Copy

```
1  @UI           : {
2        lineItem       : [{position: 15, importance: #HIG
3        selectionField: [{position: 15 }]
4        }
5  AgencyID;
6
7  @UI           : {
8        lineItem       : [{position: 20, importance: #HIG
9        selectionField: [{position: 20 }]
10       }
11 CustomerID;
```

```
12
13  @UI           : {
14        lineItem      : [{position: 10, importance: #HIG
15        selectionField: [{position: 10 }]
16        }
17  TravelID;
18
19  @UI           : {
20        lineItem      : [{position: 30, importance: #HIG
21        selectionField: [{position: 30 }]
22        }
23  BeginDate;
24
25  @UI           : {
26        lineItem      : [{position: 40, importance: #HIG
27        selectionField: [{position: 40 }]
28        }
29  EndDate;
30
31  @UI           : {
32        lineItem      : [{position: 50, importance: #HIG
33        }
34  TotalPrice;
35
36  @UI           : {
37        lineItem      : [{position: 50, importance: #HIG
38        }
39  Memo;
40
41  @UI           : {
42      lineItem      : [{position: 60, importance: #HIGH}
43      selectionField: [{position: 60 }]
44      }
45  Status;
```

Your app should now look like this:



Done

Log on to answer question

14 Step 14: Check your code

Your CDS entity code should look like this:

CDS Copy

```
@AbapCatalog.sqlViewName: 'ZCTRAVEL_XXX'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Consumption view from /DMO/I_TRAVEL_U'
@Metadata.allowExtensions: true
@Search.searchable: true

```

```
define view Z_C_TRAVEL_DATA_XXX
  as select from /DMO/I_Travel_U
{

      ///DMO/I_Travel_U

  key TravelID,
      AgencyID,
      CustomerID,
      BeginDate,
      EndDate,
      BookingFee,

      @Semantics.amount.currencyCode: 'CurrencyCode'
      TotalPrice,

      @Semantics.currencyCode
      CurrencyCode,

      @Search.defaultSearchElement: true
      @Search.fuzzinessThreshold: 0.90
      Memo,
      Status,
      LastChangedAt,

      /* Associations */
      ///DMO/I_Travel_U
      _Agency,
      _Booking,
      _Currency,
      _Customer

}

```

Your MDE code should look like this:

CDS Copy

```
@Metadata.layer: #CORE
annotate view Z_C_TRAVEL_DATA_XXX with
{

@UI           : {
      lineItem      : [{position: 15, importance: #HIGH}],
      selectionField: [{position: 15 }]
      }
AgencyID;

@UI           : {
      lineItem      : [{position: 20, importance: #HIGH}],
      selectionField: [{position: 20 }]
      }
CustomerID;

@UI           : {
      lineItem      : [{position: 10, importance: #HIGH}],
      selectionField: [{position: 10 }]
      }
TravelID;

@UI           : {
      lineItem      : [{position: 30, importance: #HIGH}],
      selectionField: [{position: 30 }]
      }
BeginDate;

@UI           : {
```

```
30        lineItem      : [{position: 40, importance: #HIGH}],
31        selectionField: [{position: 40 }]
32        }
33  EndDate;
34
35  @UI           : {
36        lineItem      : [{position: 50, importance: #HIGH}]
37        }
38  TotalPrice;
39
40  @UI           : {
41        lineItem      : [{position: 50, importance: #HIGH}]
42        }
43  Memo;
44
45  @UI           : {
46      lineItem      : [{position: 60, importance: #HIGH}],
47      selectionField: [{position: 60 }]
48      }
49  Status;
50
51  }
52
```

Done

Log on to answer question

Step 15: Test yourself

Where do you specify that your OData service is an OData version 2 service for UI?

- ◯ Data definition
- ◯ Metadata extension
- ◯ Service definition
- ◯ Service binding

Submit Answer

Log on to answer question ×

Next Steps

This tutorial is part of these groups and missions:

Group
Beginner
1 hr. 50 min.

2 tutorials

Create and Enhance a CDS View in SAP Cloud Platform, ABAP Environment

Create and enhance a CDS view using annotations and expressions, to make your applications more powerful and more user-friendly.

ABAP Development

Developer Products

ABAP Platform

SAP Business Application Studio

SAP Business Technology Platform

SAP Conversational AI

SAP Data Intelligence

SAP HANA

All Products

Trials & Downloads

ABAP Development Tools

Mobile Development Kit Client

SAP Business Application Studio

SAP Data Intelligence Trial

SAP HANA Cloud Trial

All Trials & Downloads

Site Information

Privacy

Terms of Use

Legal Disclosure

Copyright

Trademark

Newsletter

Sitemap

Text View

Find us on