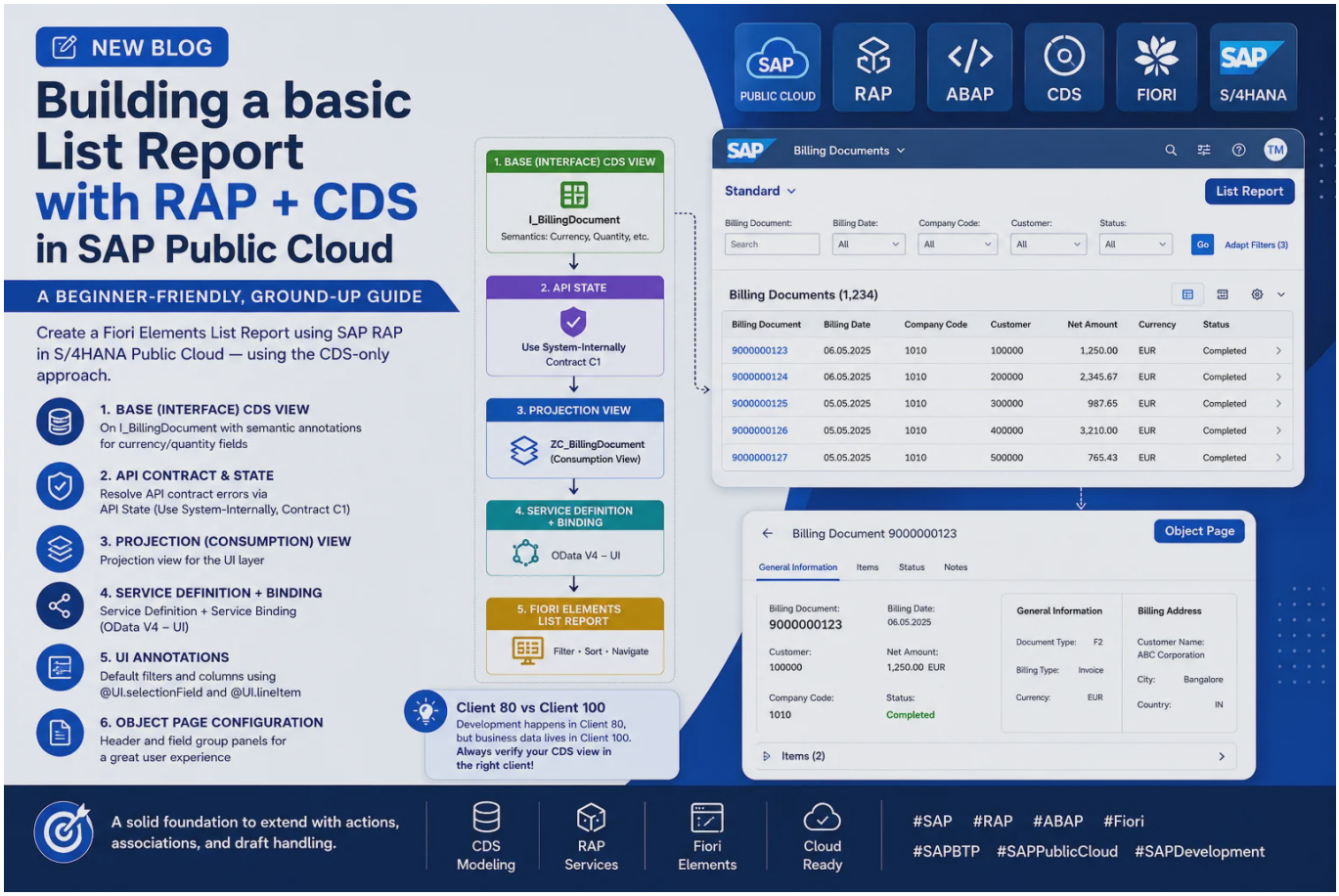
SAP RAP Basic List Report Using CDS in Public Cloud

June 27, 2026 | SAP, RAP, ABAP, List Report, Public Cloud



Introduction

List Reports are among the most widely used SAP Fiori application patterns for displaying and managing business data in a structured, tabular format. Built on the SAP Fiori Elements framework, they deliver a consistent and intuitive user experience while dramatically reducing the need for custom UI development.

In SAP S/4HANA Public Cloud, the SAP RAP (RESTful ABAP Programming Model) empowers developers to build enterprise-grade applications using CDS views, service definitions, and service bindings. With just a handful of development artifacts, you can generate a fully functional List Report that supports standard features such as filtering, sorting, free-text search, column personalization, variant management, and seamless navigation.

In this tutorial, you will learn how to build a basic List Report from scratch. We will walk through creating the required CDS data model, defining the business object, exposing it as an OData service, and previewing a SAP Fiori Elements List Report application.

By the end of this guide, you will have a working List Report running in your SAP S/4HANA Public Cloud system, along with a solid foundation for developing more advanced RAP applications.

Approaches to Creating a List Report

There are two common approaches to creating a List Report in SAP RAP:

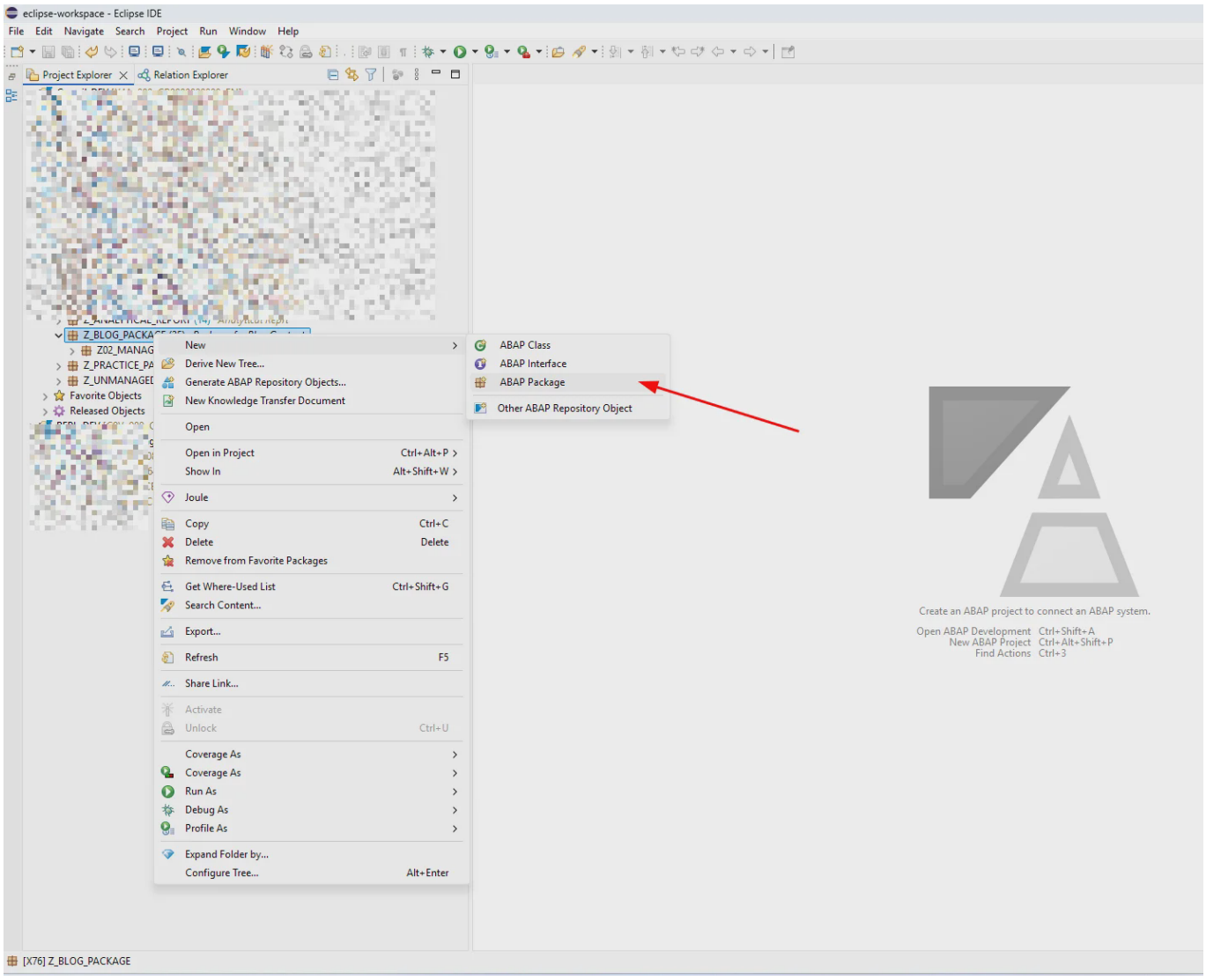
- **Using CDS Views Only** -- The data retrieval logic is defined entirely within CDS views. This is the recommended approach for most standard scenarios, as it requires less custom code and fully leverages SAP's optimized framework.
- **Using an ABAP Class as a Query Provider** -- A custom ABAP class implements the query logic, acting as the data provider. This approach is suited for scenarios where business logic is too complex to express using CDS alone.

In this tutorial, we will follow the **CDS-based approach**.

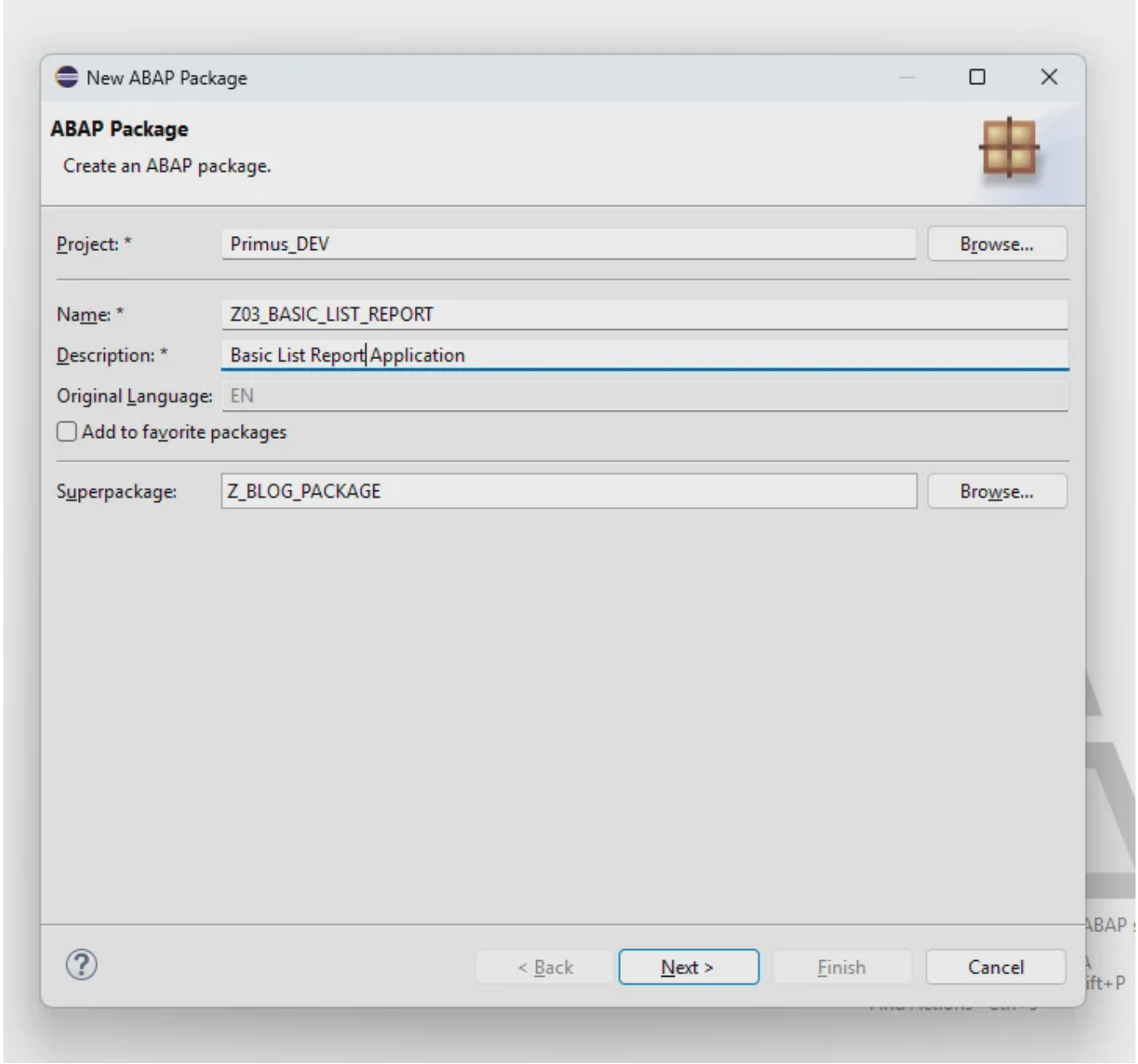
Step 1: Create a Development Package

The first step is to create a development package that will serve as the container for all objects required by this application -- including CDS views, behavior definitions, service definitions, and other RAP artifacts.

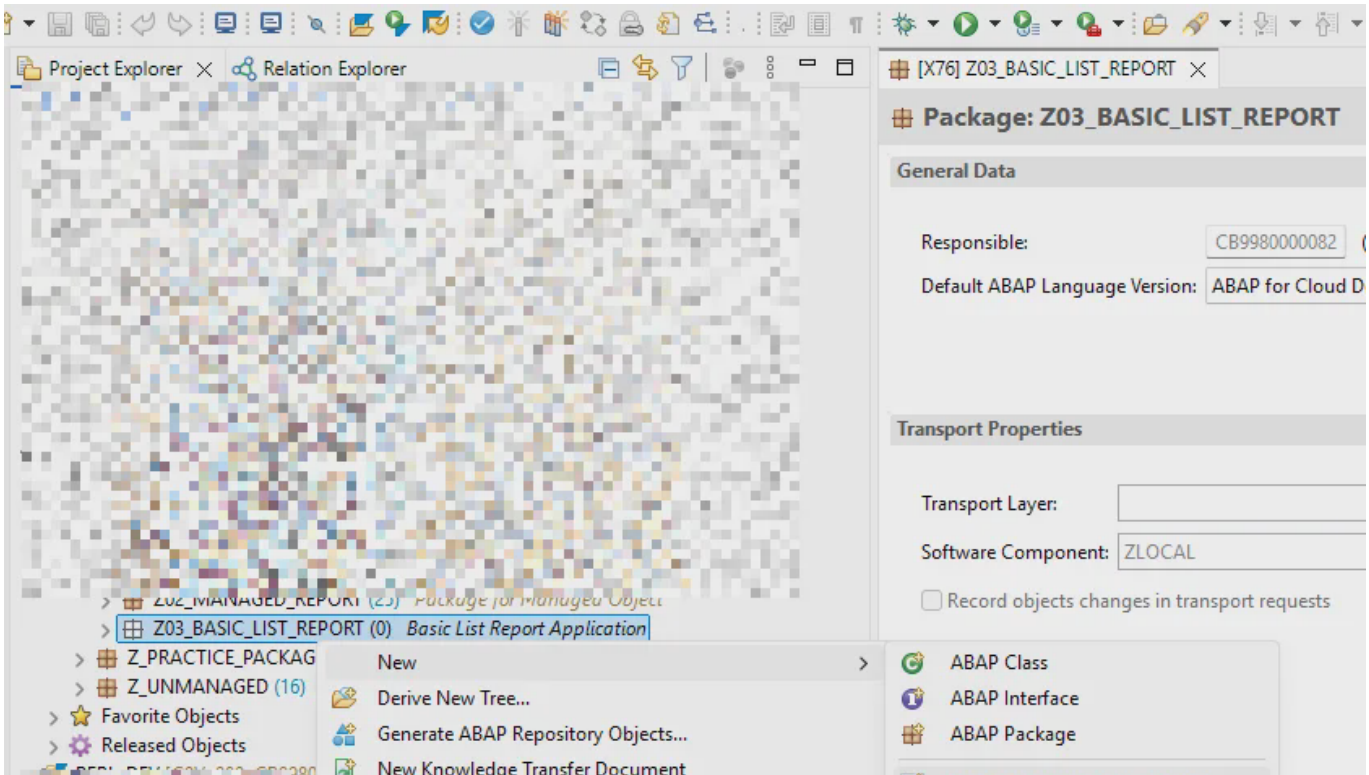
Create a new package in your development system.



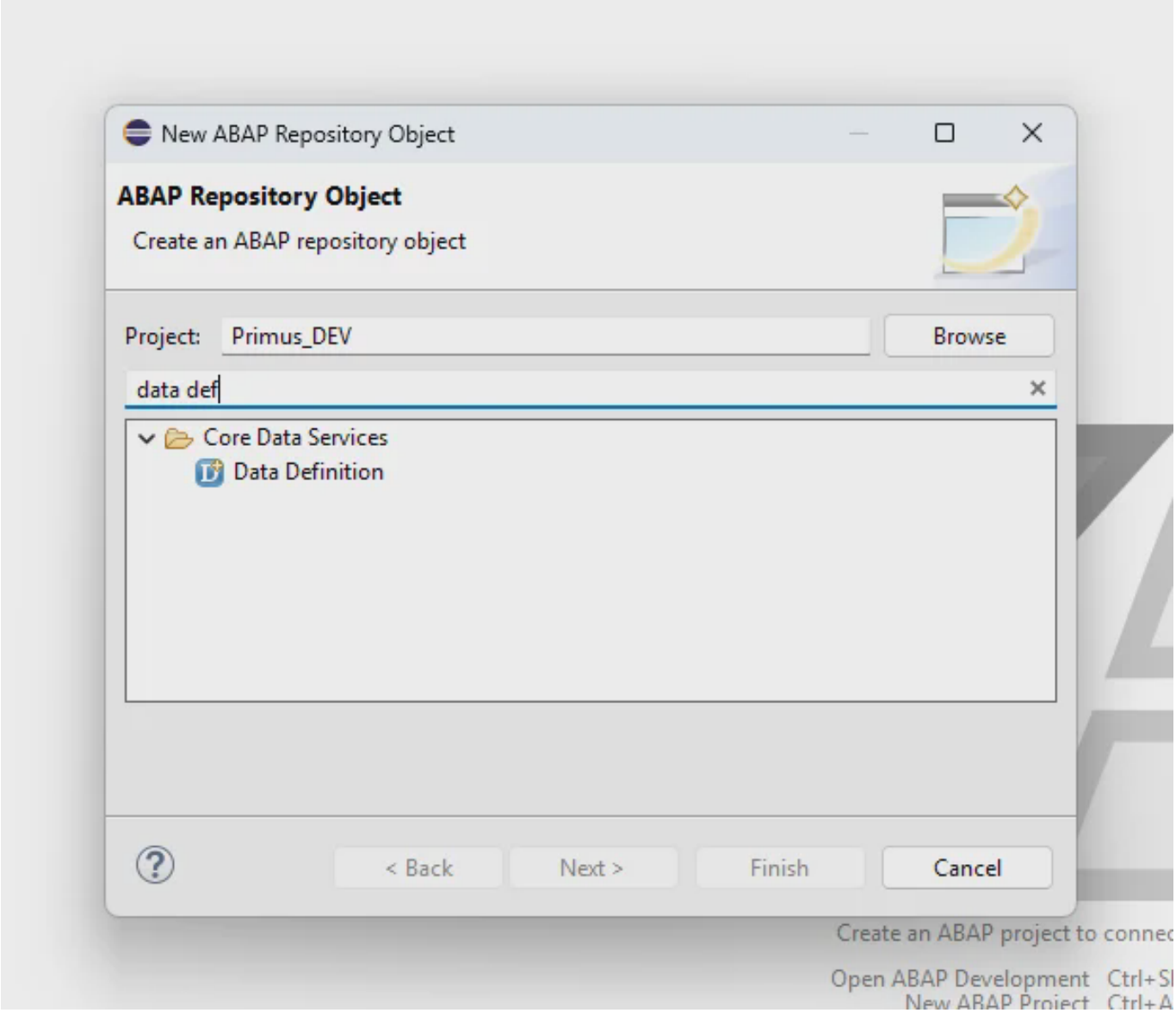
Enter the package details, then save and activate the package.



Right-click on your package and select **Other ABAP Repository Object**.

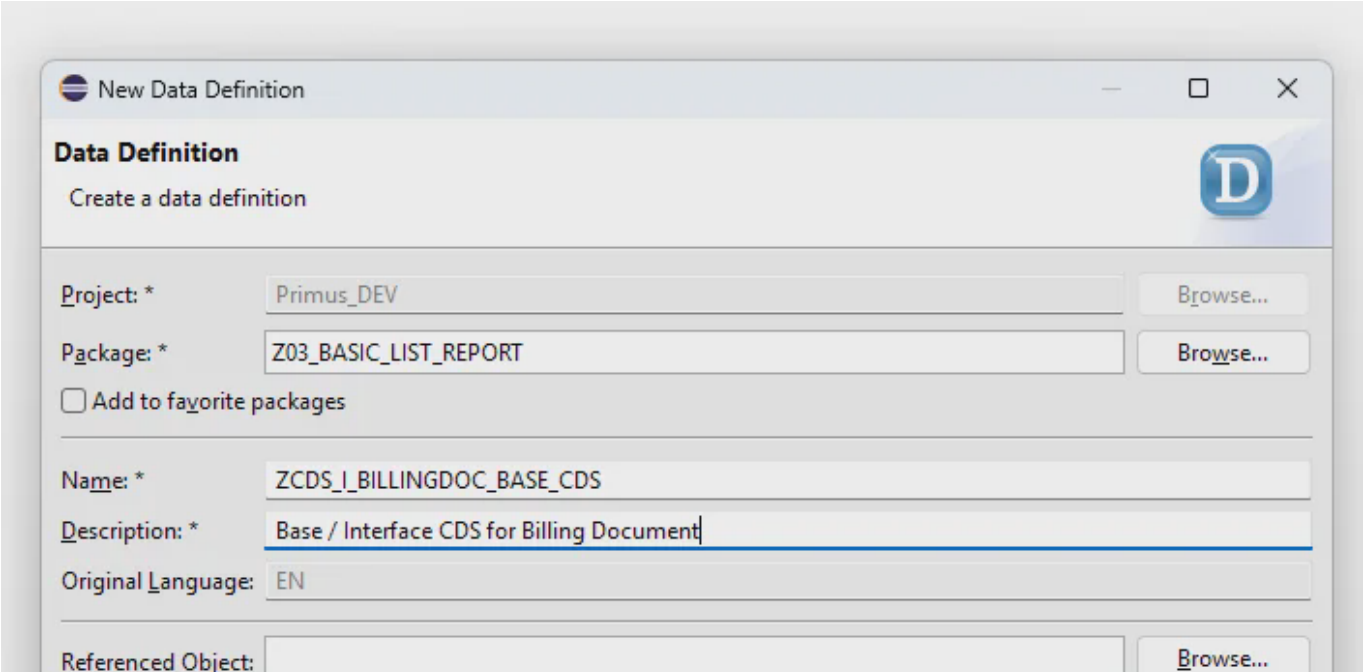


Search for **Data Definition** -- this is the object type used to create a CDS view.

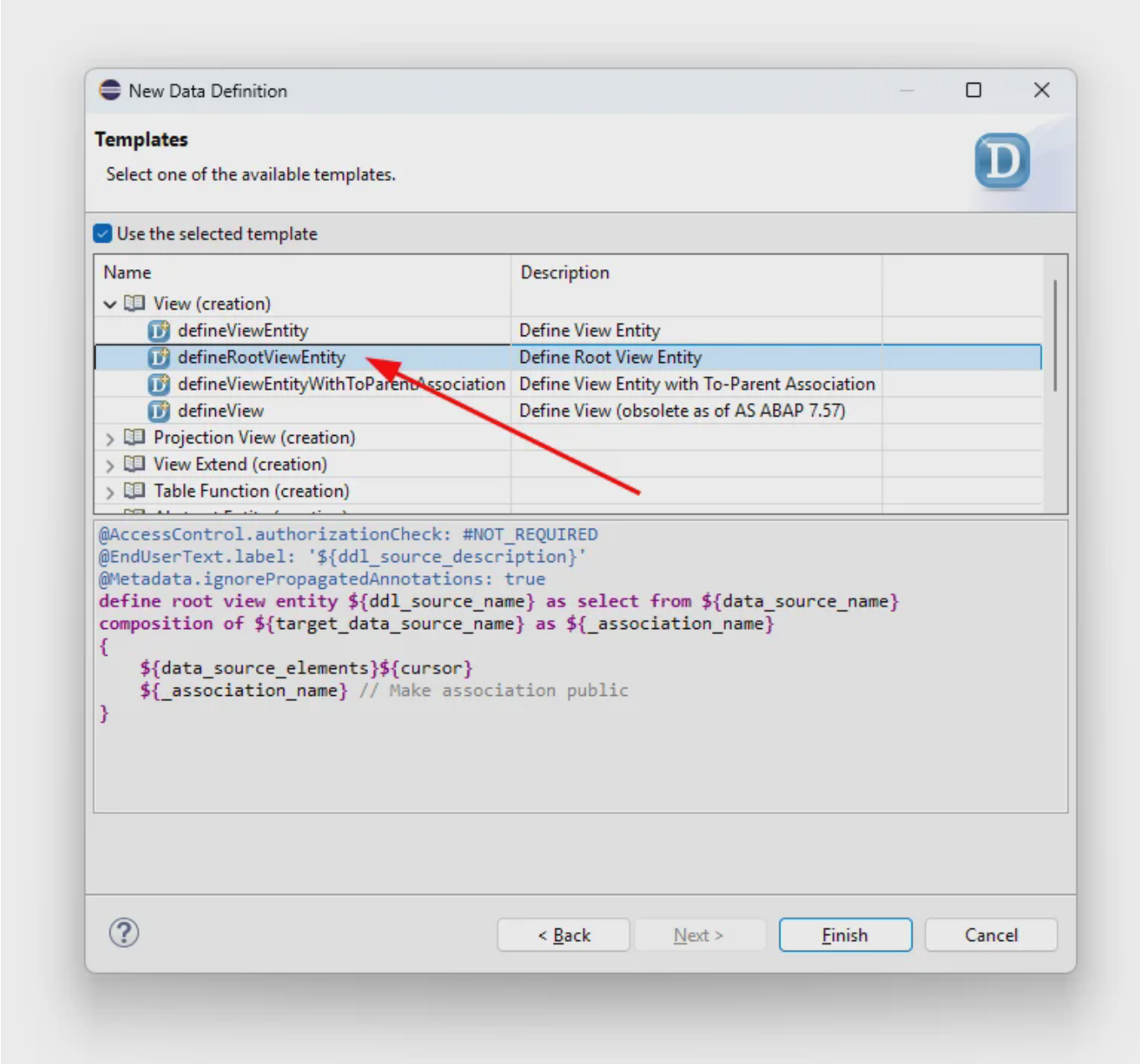


Step 2: Create the Base (Interface) CDS View

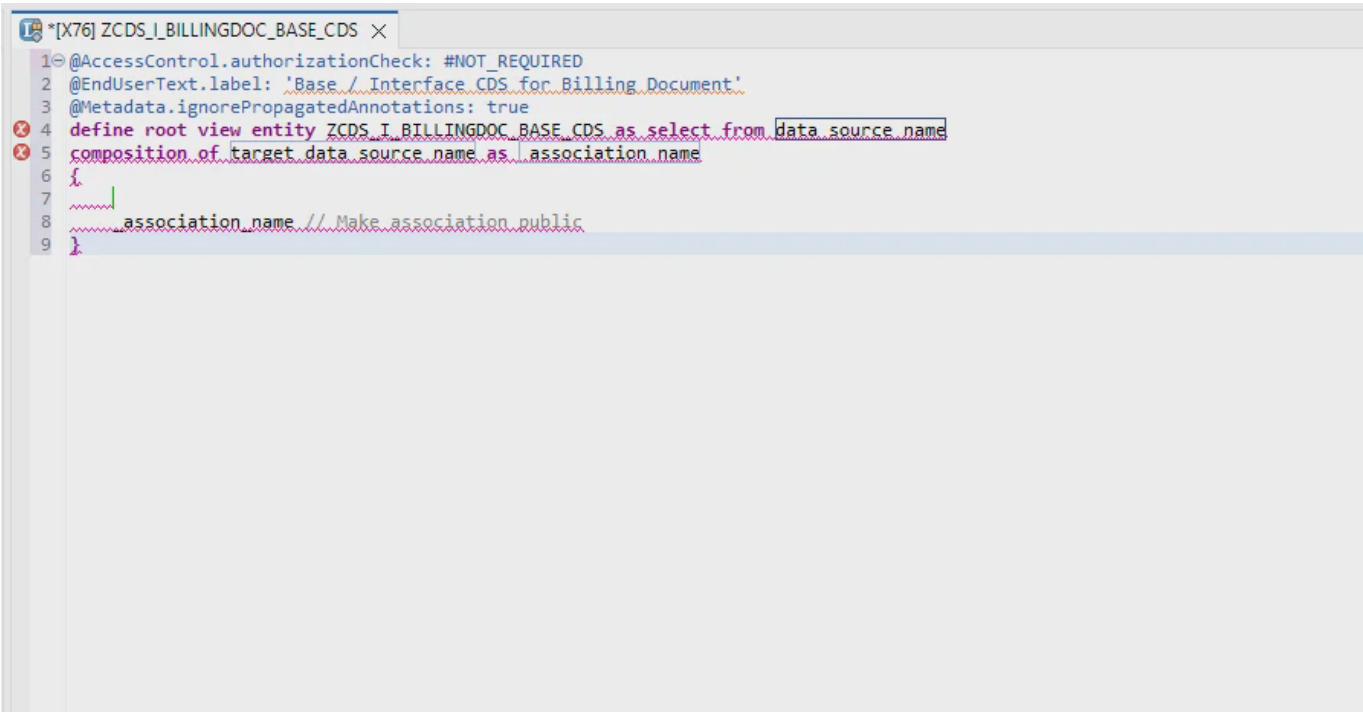
Begin by creating the Base or Interface CDS view, which serves as the foundation of your data model.



From the available CDS templates, select **defineRootViewEntity**.

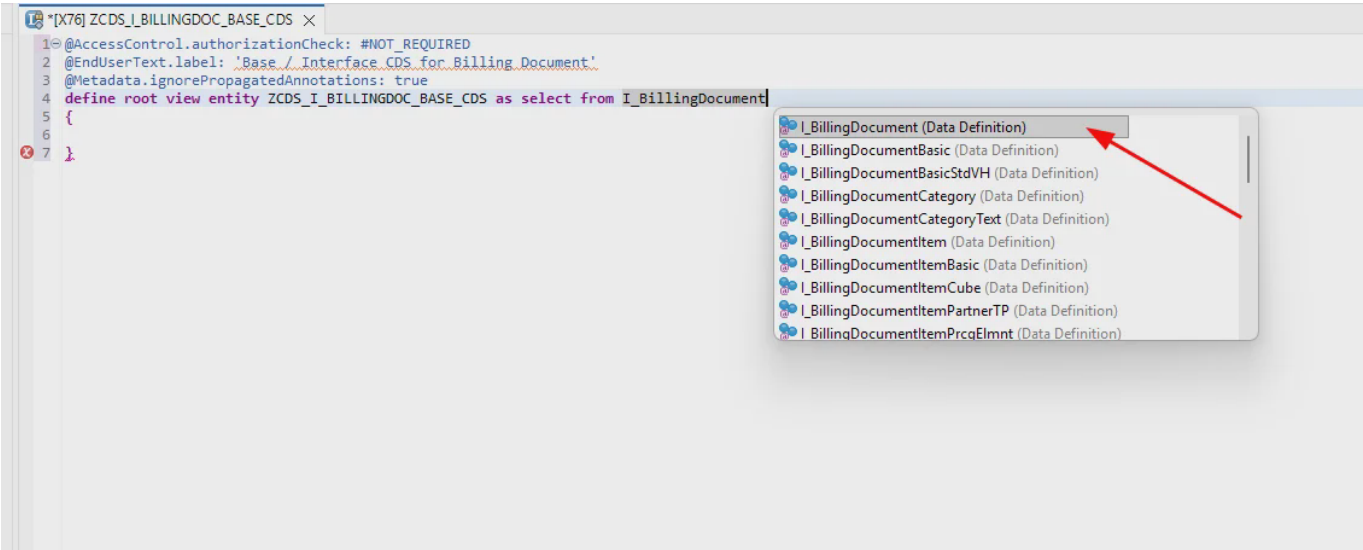


The template generates boilerplate code. Remove any unnecessary scaffolding to keep the view clean and focused.

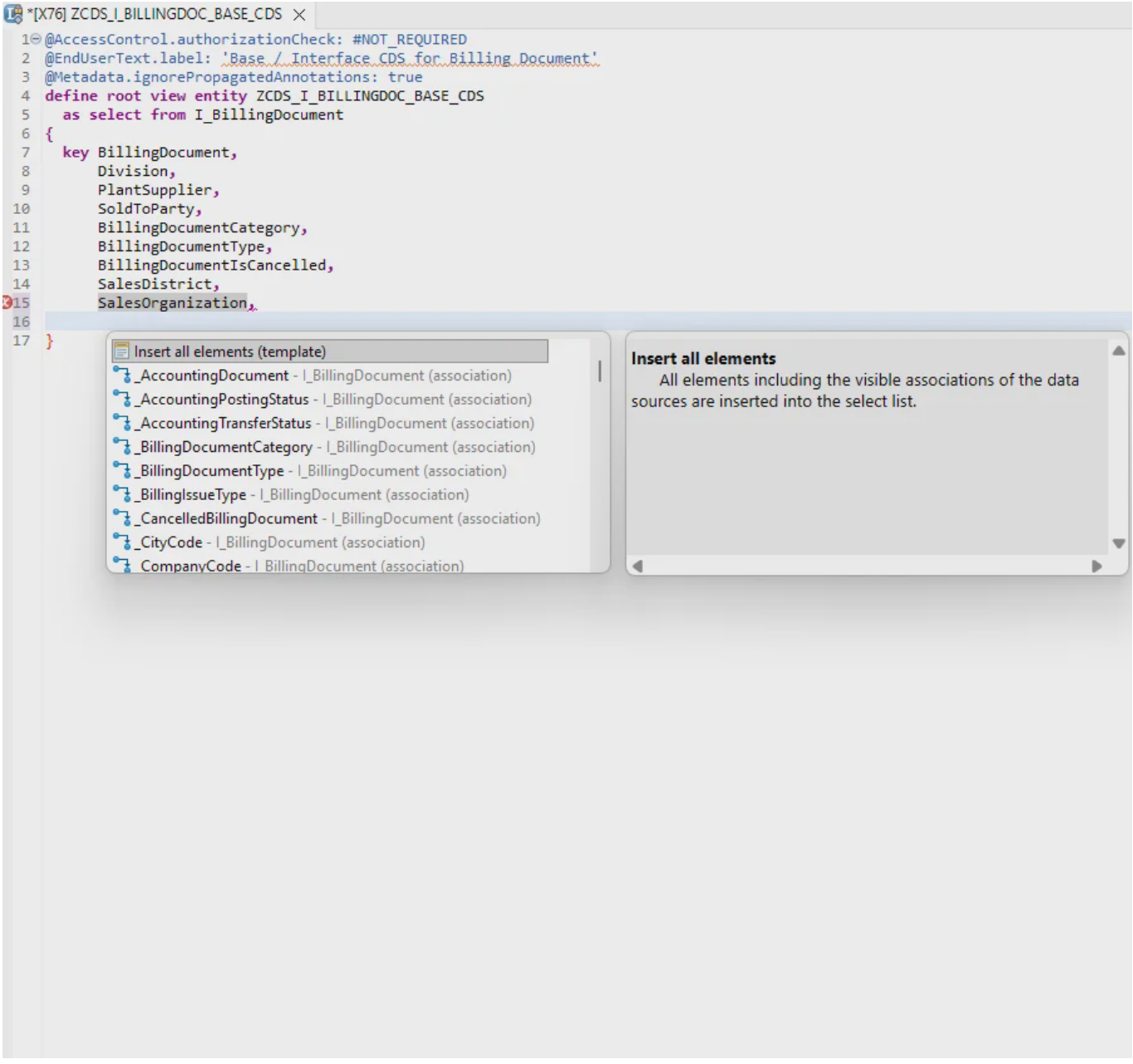


In the SELECT statement, use `I_BillingDocument` as the data source.

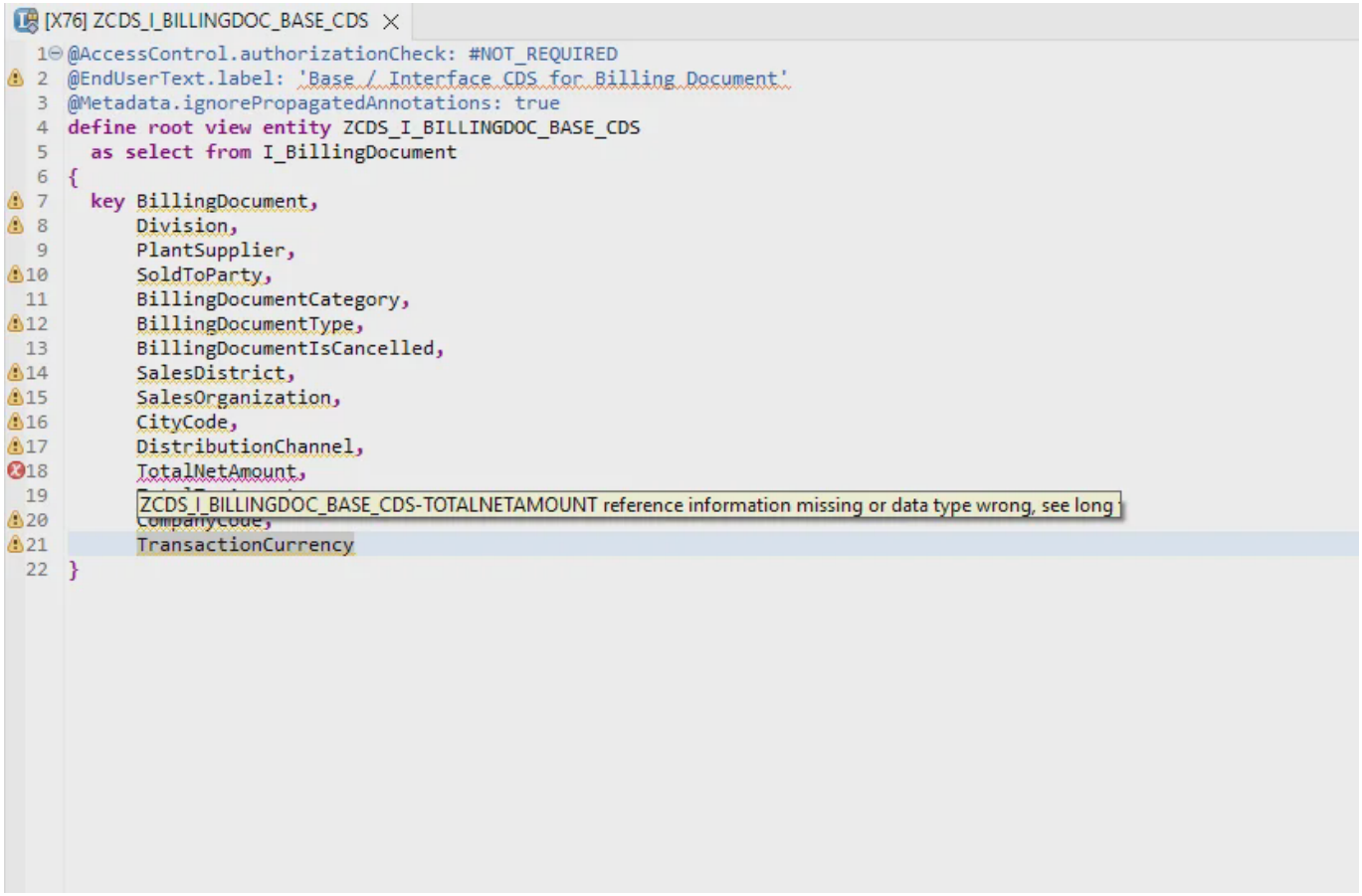
> **Tip:** Press `Ctrl + Space` to activate auto-suggestions.



Add all required fields. Use the auto-suggestion feature and select **Insert all elements (template)** to automatically pull all fields from `I_BillingDocument` .

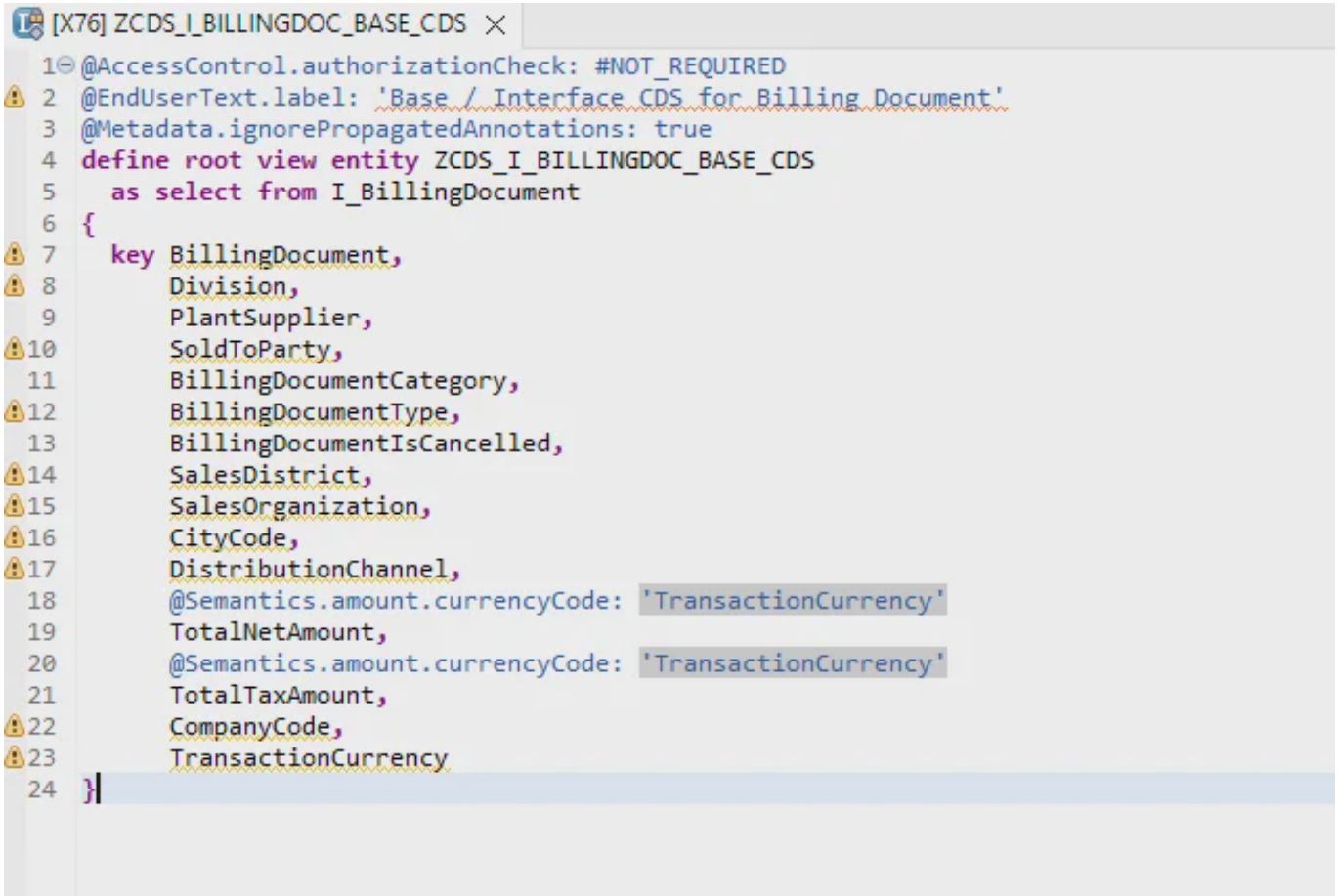


You will notice an error related to `TotalNetAmount` . This is a currency amount field, and SAP requires that all amount fields be linked to a corresponding currency field via annotations.

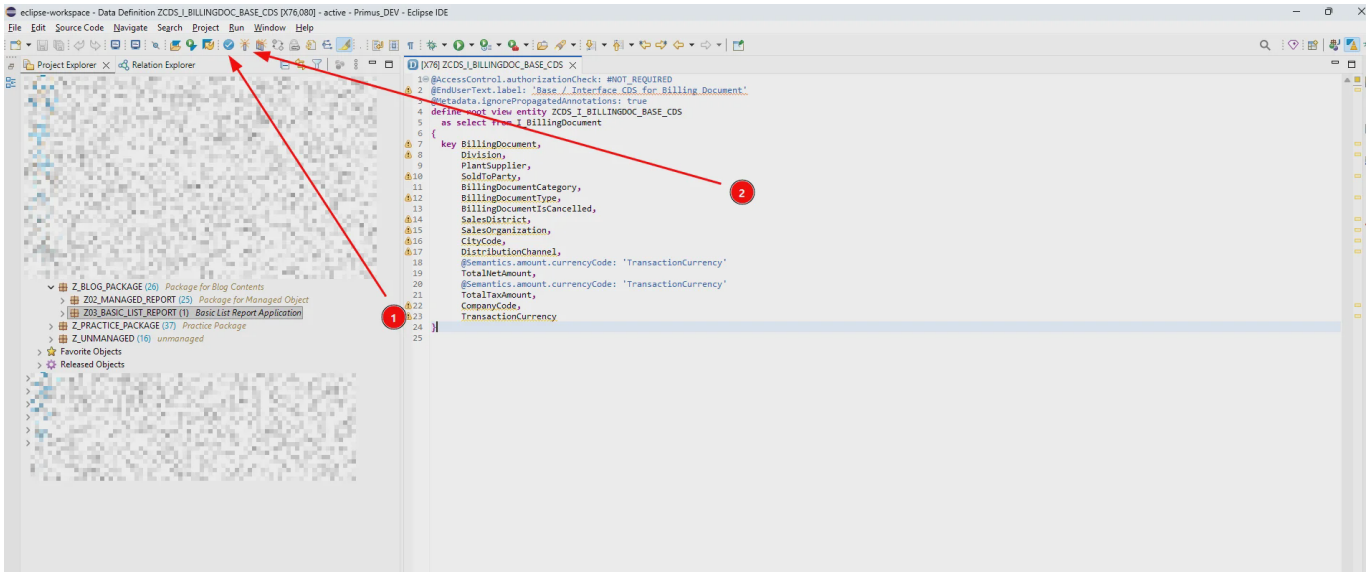


Link both `TotalNetAmount` and `TotalTaxAmount` to `TransactionCurrency` using the appropriate semantic annotation.

> **Note:** A similar error will occur for quantity fields. In those cases, use `@Semantics.quantity.unitOfMeasure: ''` and reference the corresponding unit of measure field.

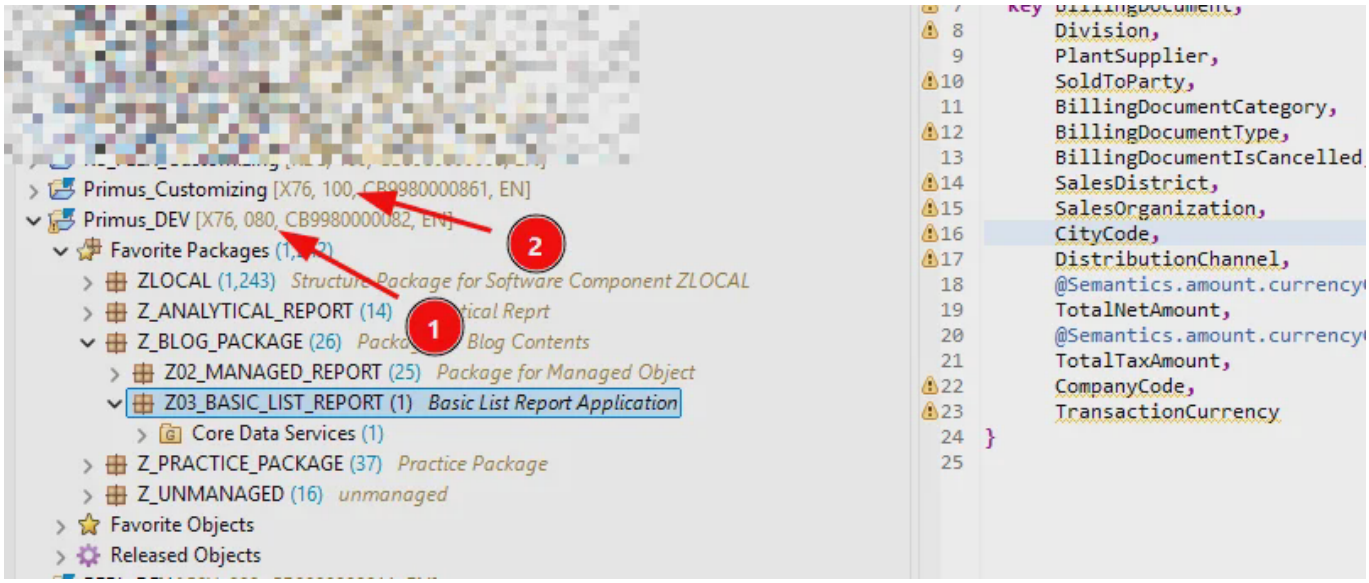


Perform a syntax check, then activate the CDS view.



Step 3: Verify Data in the Correct Client

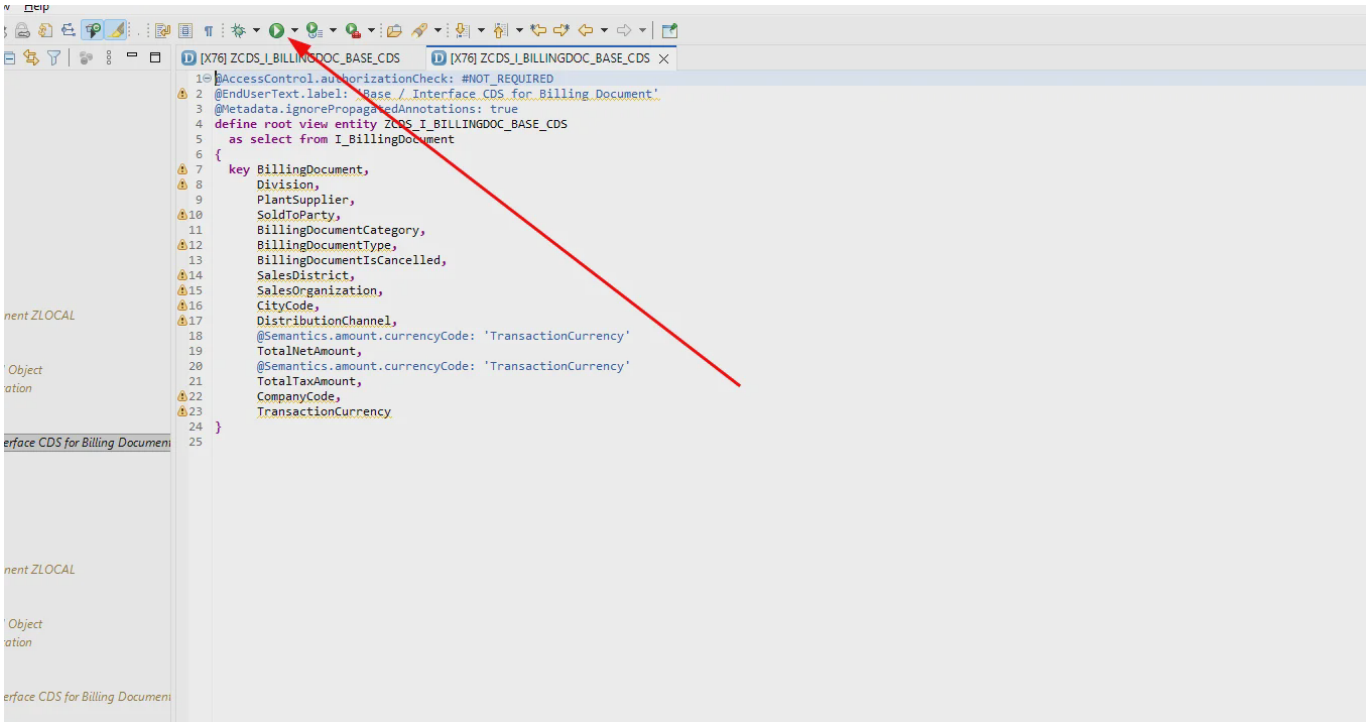
Development work takes place in **Client 80** (the development system). Business data, however, resides in **Client 100** (the production-equivalent client in Public Cloud).



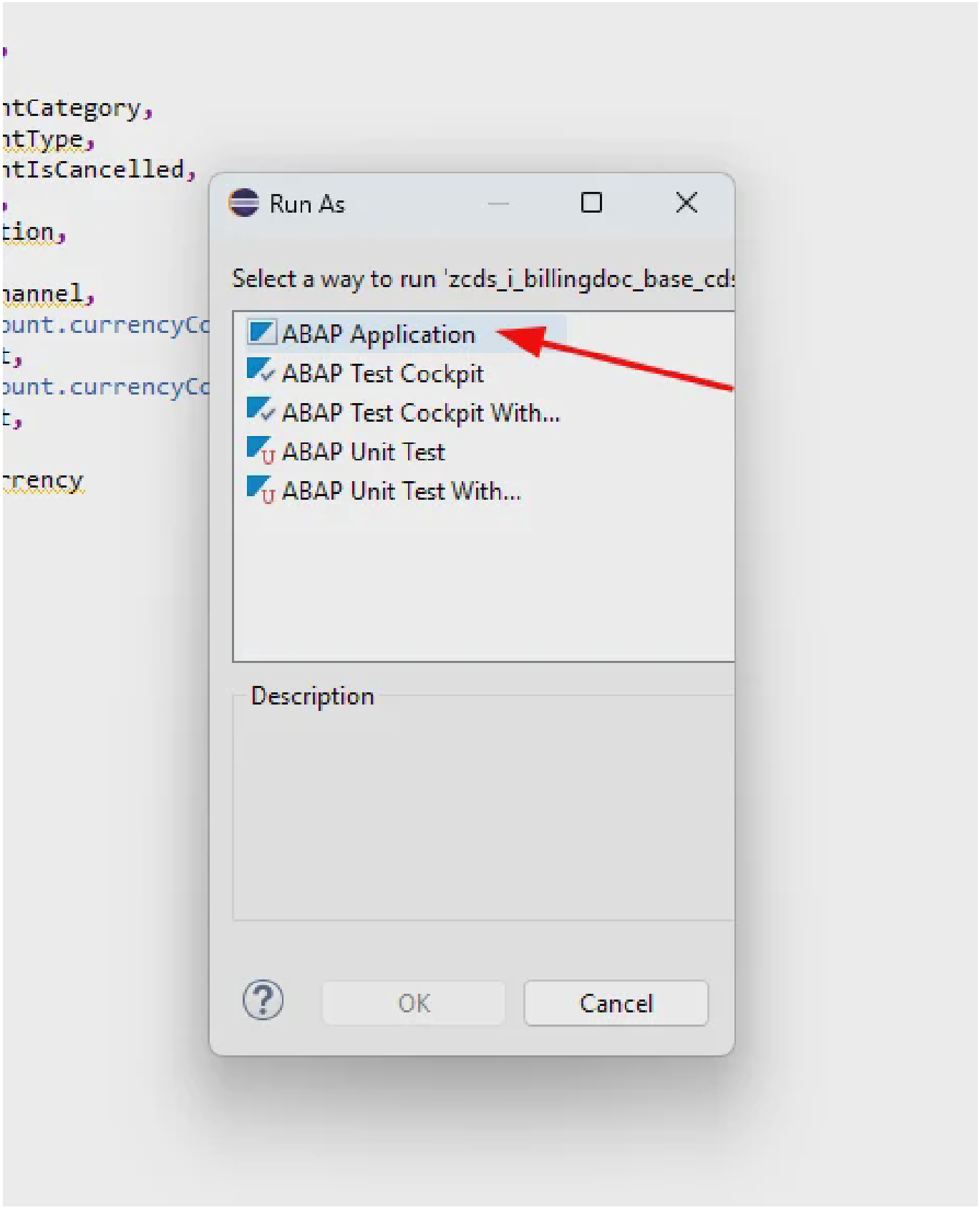
Open the same CDS view in **Client 100** to verify the data.



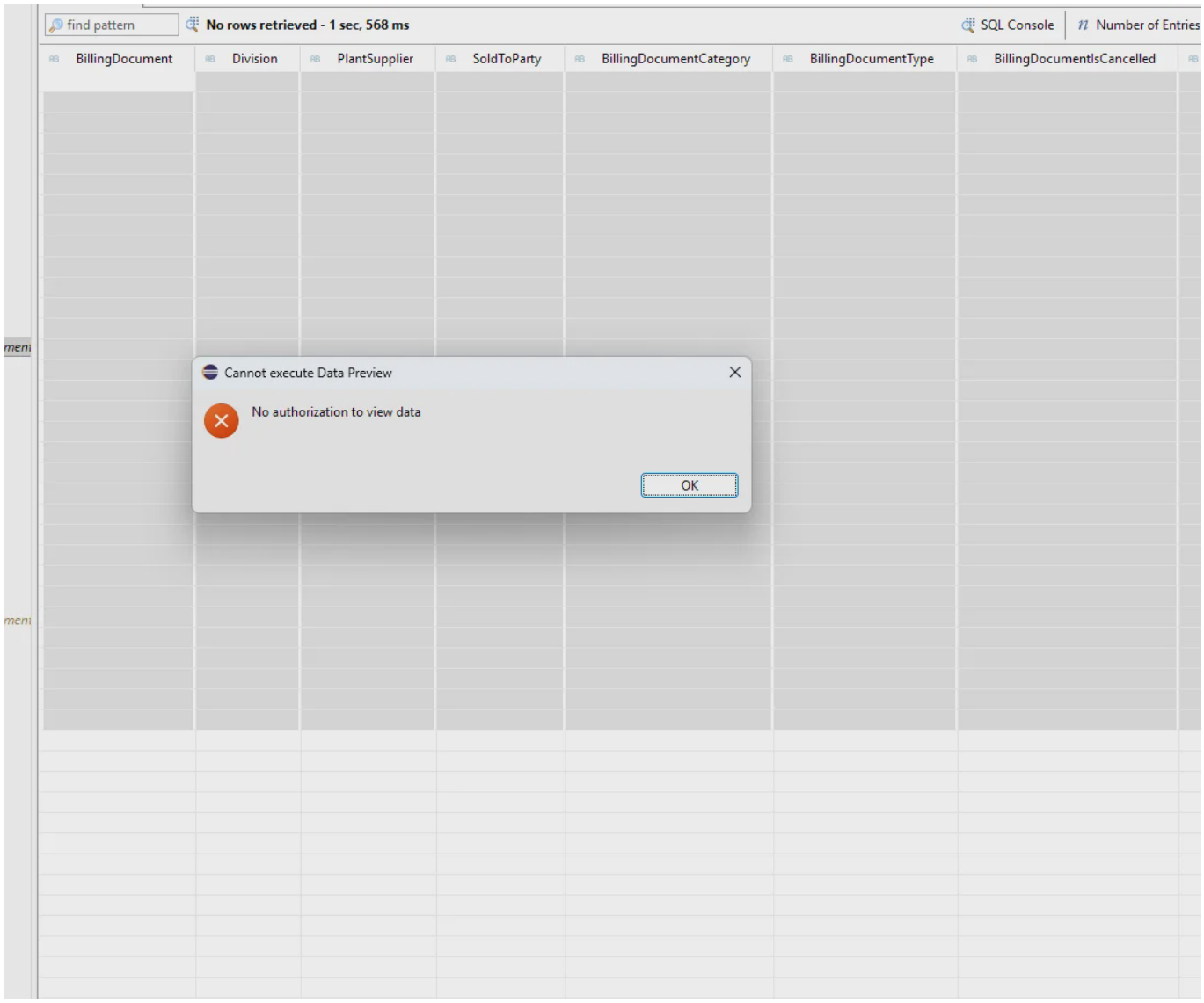
Execute the CDS view in Client 100 to preview the results.



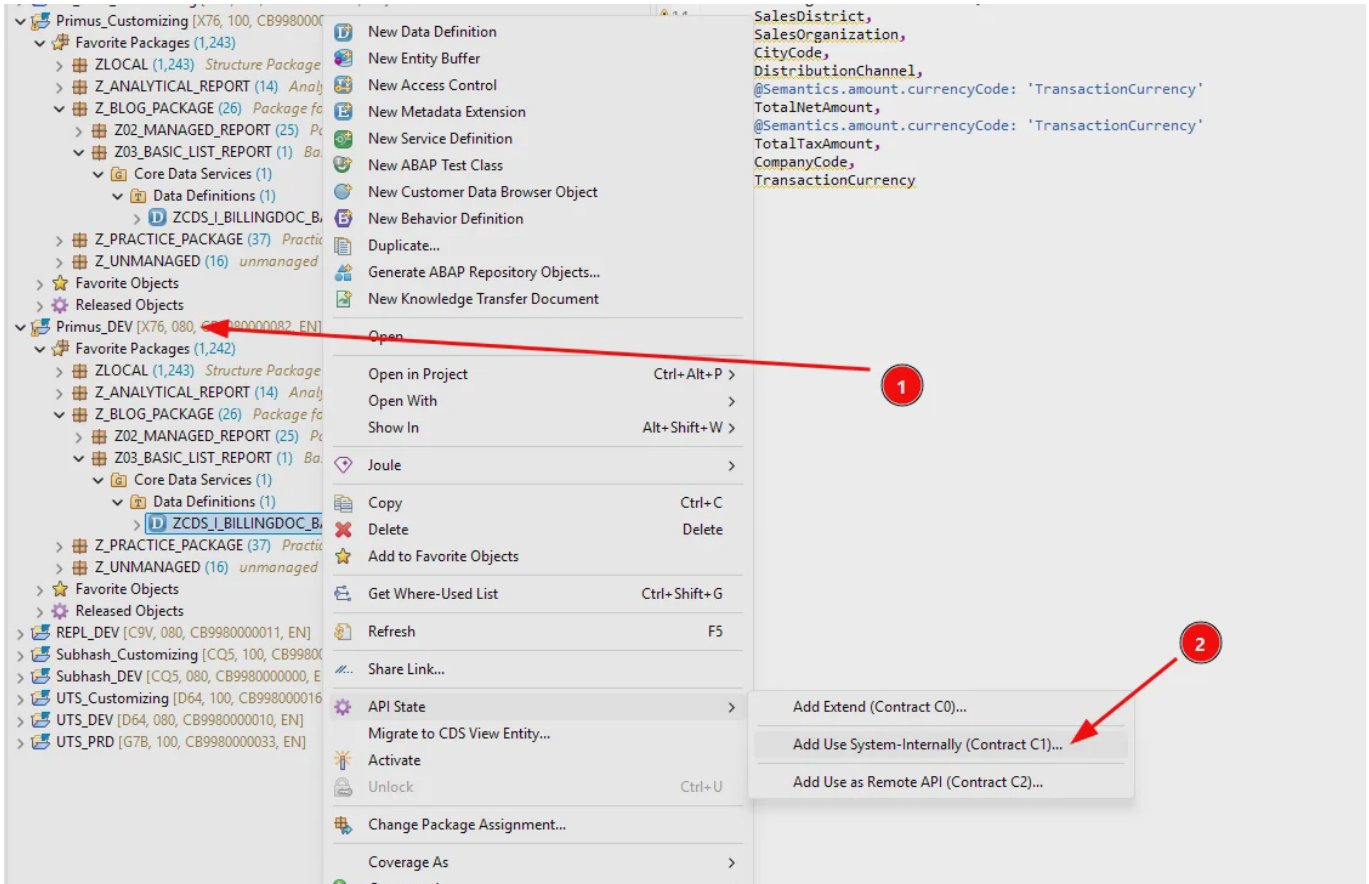
Select **ABAP Application**, or press `F8` to run the preview.



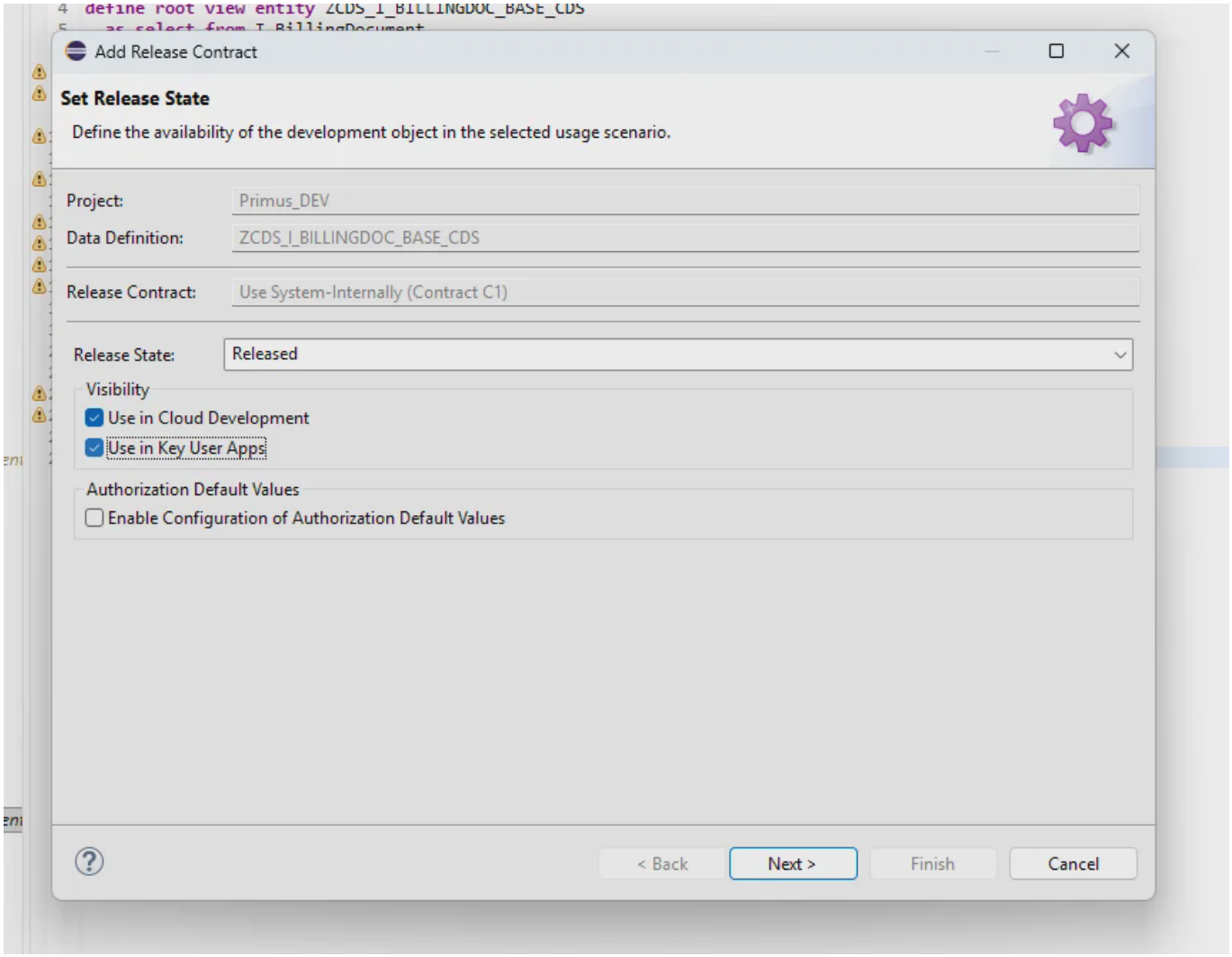
If an error appears at this point, it is typically caused by a missing API contract on the standard CDS view being consumed. To resolve this, add an **API State**.



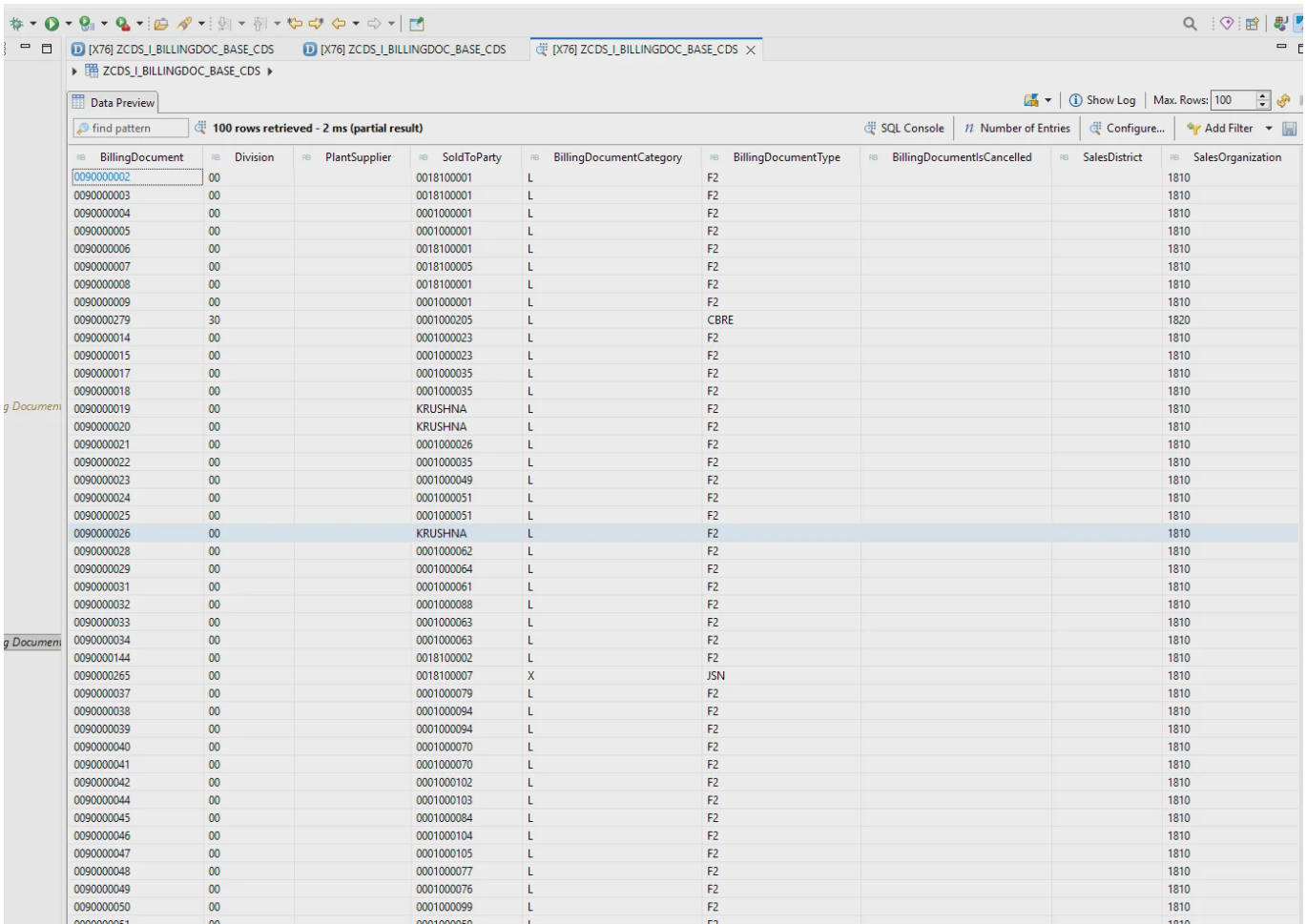
Right-click the CDS view you created in Client 80, then navigate to **API State -> Add Use System-Internally (Contract C1)**.



Check both **Use in Cloud Development** and **Use in Key User Apps**, then proceed to the next step.



After applying the API state, re-execute the CDS view. The data should now be visible.



[X76] ZCDS_I_BILLINGDOC_BASE_CDS

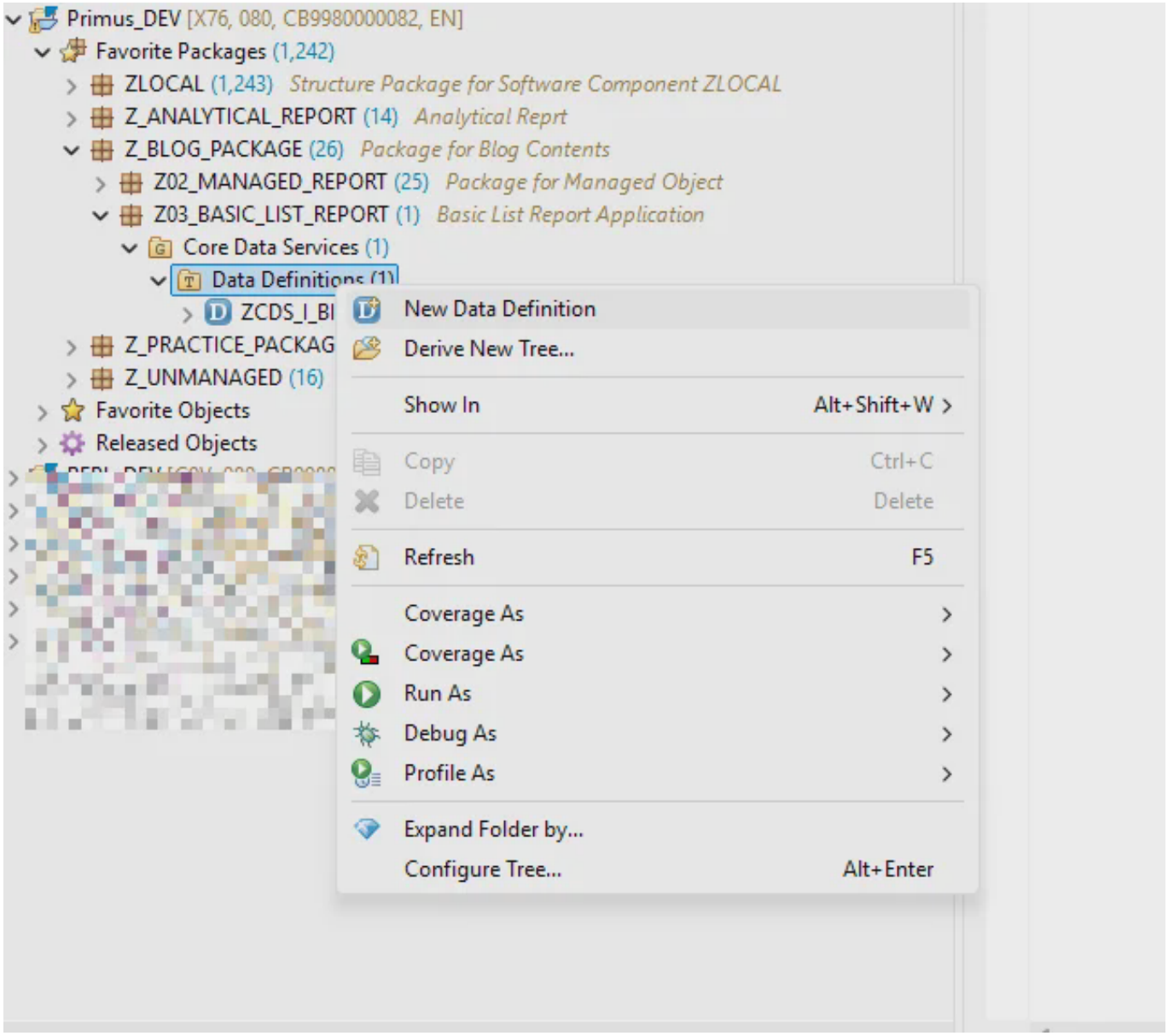
Data Preview

100 rows retrieved - 2 ms (partial result)

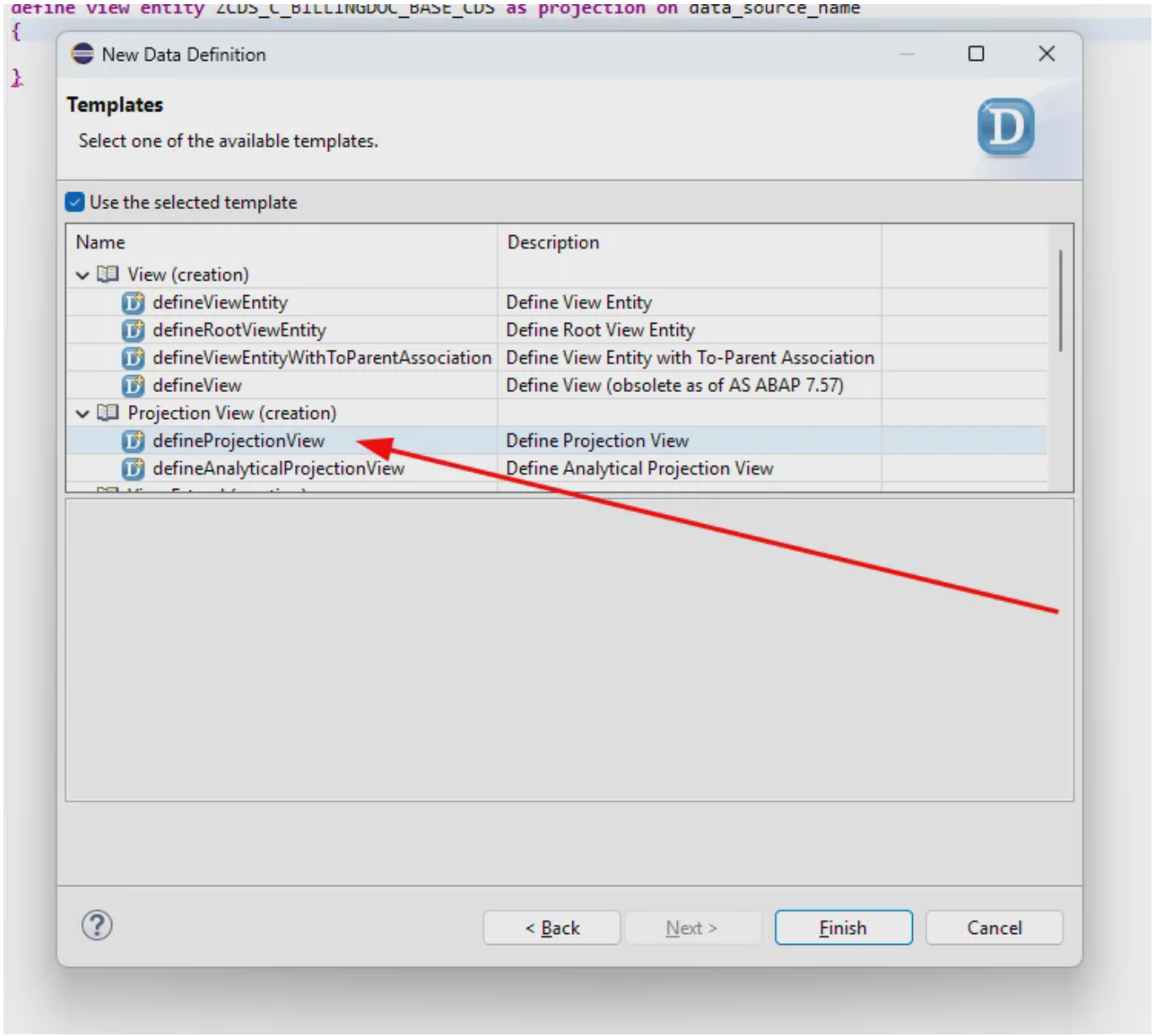
BillingDocument	Division	PlantSupplier	SoldToParty	BillingDocumentCategory	BillingDocumentType	BillingDocumentIsCancelled	SalesDistrict	SalesOrganization
0090000002	00		0018100001	L	F2			1810
0090000003	00		0018100001	L	F2			1810
0090000004	00		0001000001	L	F2			1810
0090000005	00		0001000001	L	F2			1810
0090000006	00		0018100001	L	F2			1810
0090000007	00		0018100005	L	F2			1810
0090000008	00		0018100001	L	F2			1810
0090000009	00		0001000001	L	F2			1810
0090000279	30		0001000205	L	CBRE			1820
0090000014	00		0001000023	L	F2			1810
0090000015	00		0001000023	L	F2			1810
0090000017	00		0001000035	L	F2			1810
0090000018	00		0001000035	L	F2			1810
0090000019	00		KRUSHNA	L	F2			1810
0090000020	00		KRUSHNA	L	F2			1810
0090000021	00		0001000026	L	F2			1810
0090000022	00		0001000035	L	F2			1810
0090000023	00		0001000049	L	F2			1810
0090000024	00		0001000051	L	F2			1810
0090000025	00		0001000051	L	F2			1810
0090000026	00		KRUSHNA	L	F2			1810
0090000028	00		0001000062	L	F2			1810
0090000029	00		0001000064	L	F2			1810
0090000031	00		0001000061	L	F2			1810
0090000032	00		0001000088	L	F2			1810
0090000033	00		0001000063	L	F2			1810
0090000034	00		0001000063	L	F2			1810
0090000144	00		0018100002	L	F2			1810
0090000265	00		0018100007	X	JSN			1810
0090000037	00		0001000079	L	F2			1810
0090000038	00		0001000094	L	F2			1810
0090000039	00		0001000094	L	F2			1810
0090000040	00		0001000070	L	F2			1810
0090000041	00		0001000070	L	F2			1810
0090000042	00		0001000102	L	F2			1810
0090000044	00		0001000103	L	F2			1810
0090000045	00		0001000084	L	F2			1810
0090000046	00		0001000104	L	F2			1810
0090000047	00		0001000105	L	F2			1810
0090000048	00		0001000077	L	F2			1810
0090000049	00		0001000076	L	F2			1810
0090000050	00		0001000099	L	F2			1810

Step 4: Create the Projection (Consumption) View

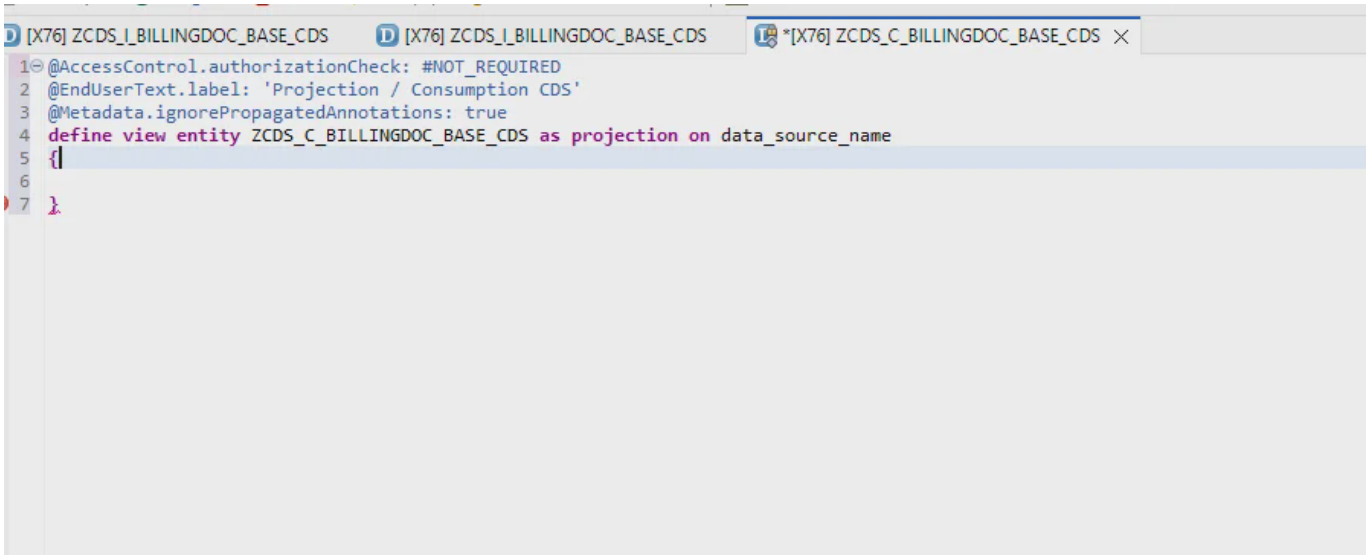
With the base CDS view in place, the next step is to create a **Projection View**, also known as a Consumption View. This layer adds UI-specific metadata and restricts the fields exposed to the frontend.



This time, select the **defineProjectionView** template.



Modify the generated boilerplate to suit your requirements.

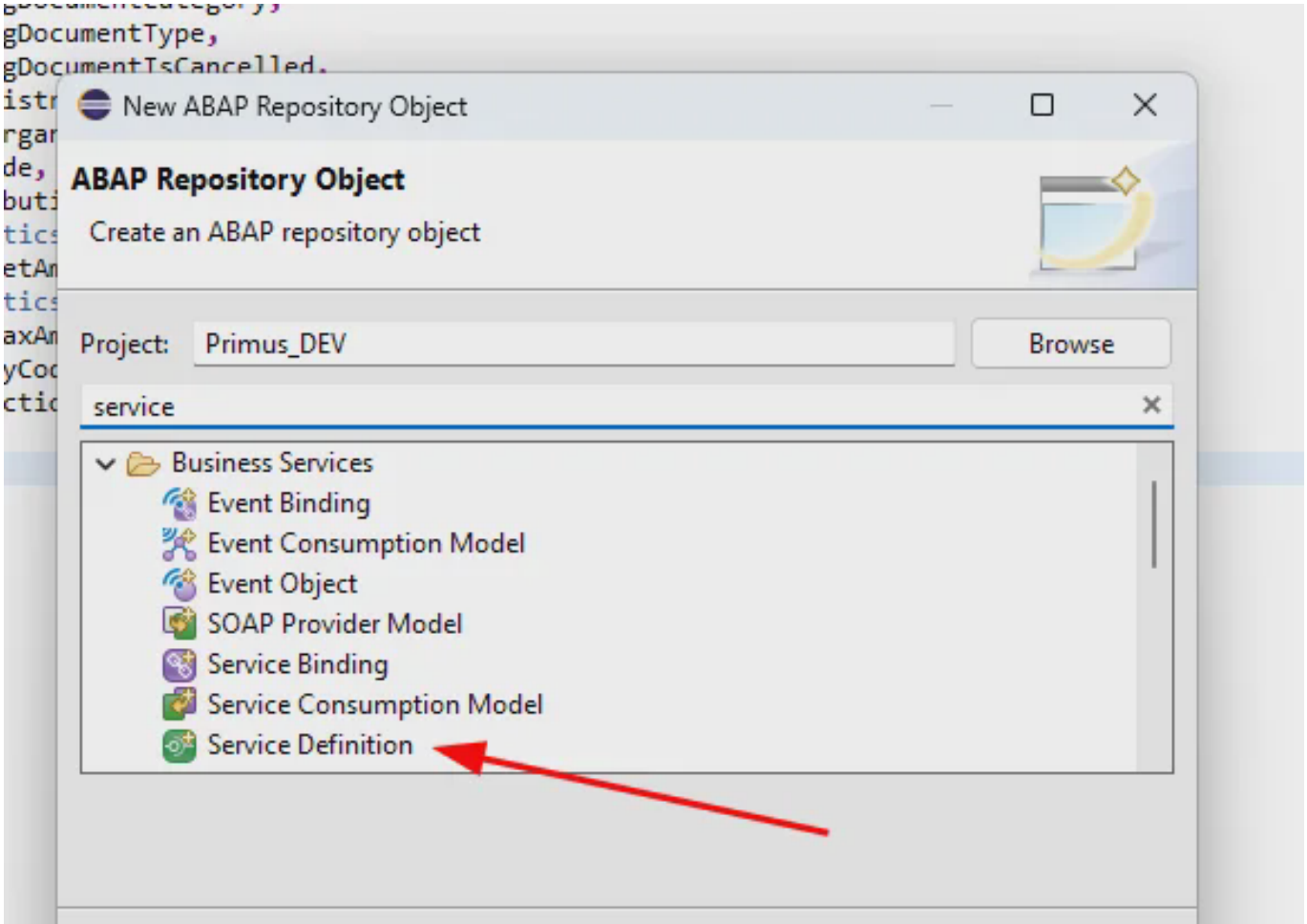


In the `PROJECTION ON` clause, reference your previously created base CDS view. Add the `root` keyword after `define` and before `view entity`, since the base view is defined as a root view entity. Then insert all elements from the base CDS view.

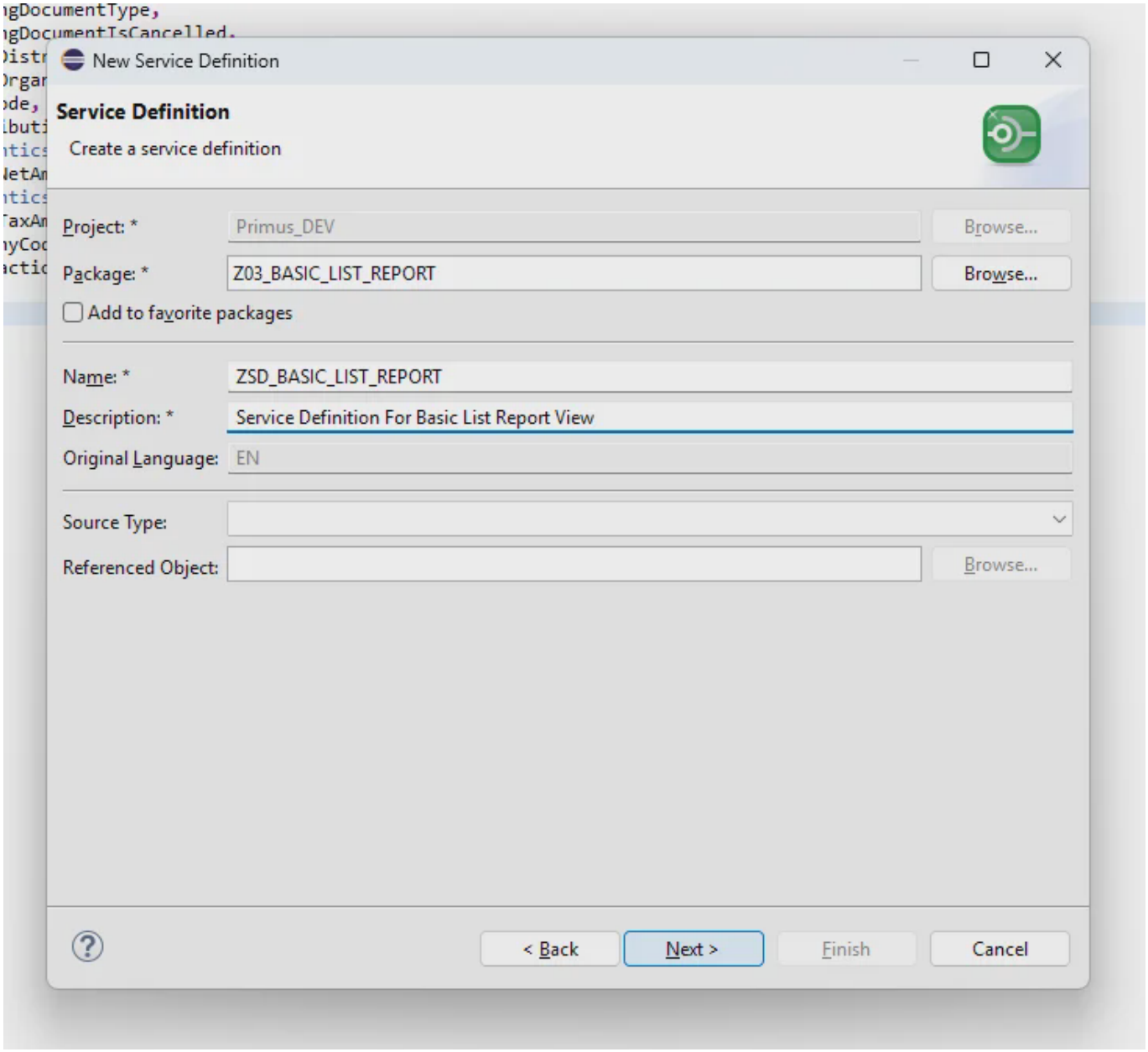
```
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Projection / Consumption CDS'
@Metadata.ignorePropagatedAnnotations: true
define root view entity ZCDS_C_BILLINGDOC_BASE_CDS
  as projection on ZCDS_I_BILLINGDOC_BASE_CDS
{
  key BillingDocument,
      Division,
      PlantSupplier,
      SoldToParty,
      BillingDocumentCategory,
      BillingDocumentType,
      BillingDocumentIsCancelled,
      SalesDistrict,
      SalesOrganization,
      CityCode,
      DistributionChannel,
      @Semantics.amount.currencyCode: 'TransactionCurrency'
      TotalNetAmount,
      @Semantics.amount.currencyCode: 'TransactionCurrency'
      TotalTaxAmount,
      CompanyCode,
      TransactionCurrency
}
```

Step 5: Create the Service Definition

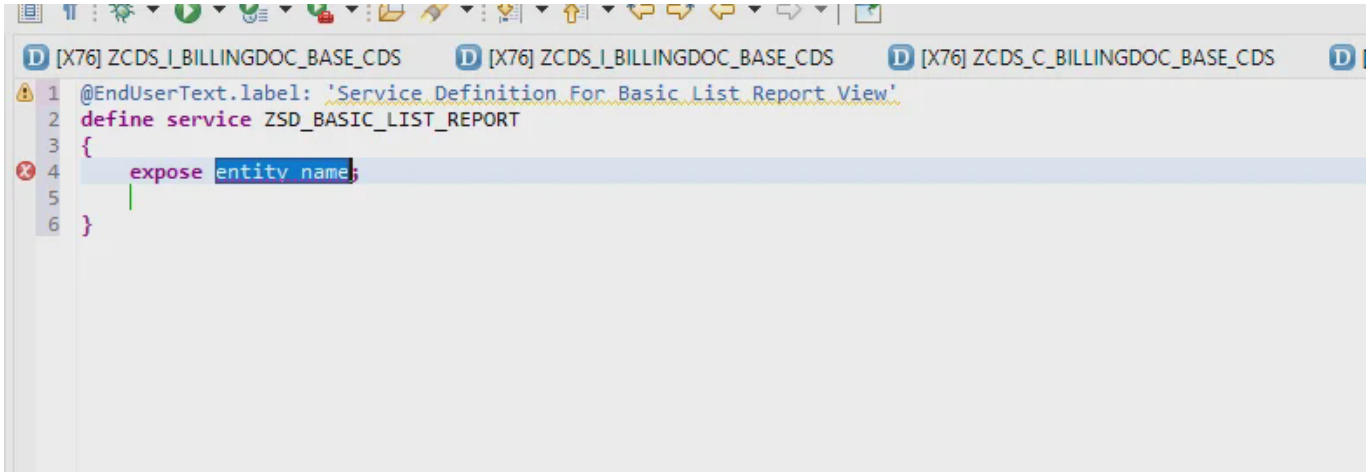
The Service Definition exposes your projection view as an OData service entity.



Enter the required name and description.



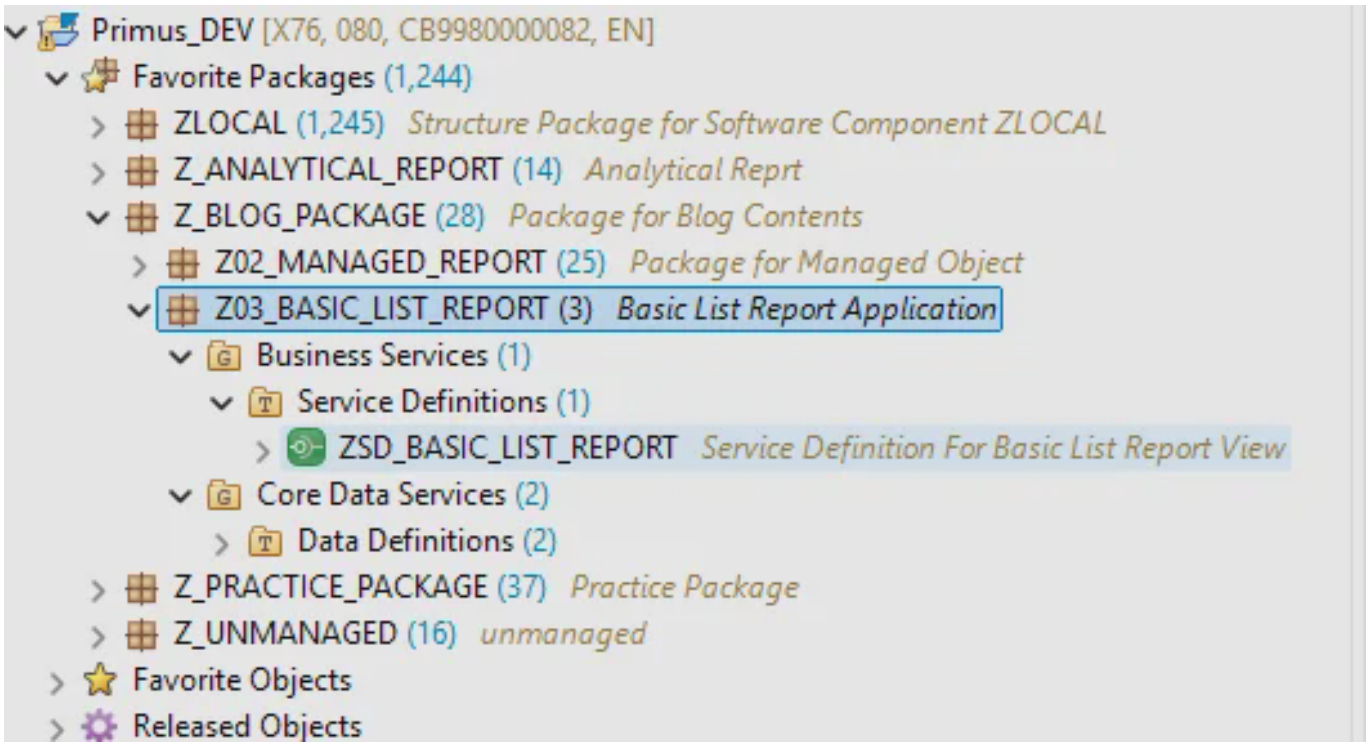
The generated boilerplate will look like this.



Expose your Projection View with a meaningful alias using the `as` keyword. In this example, the alias is set to `BillingList` .

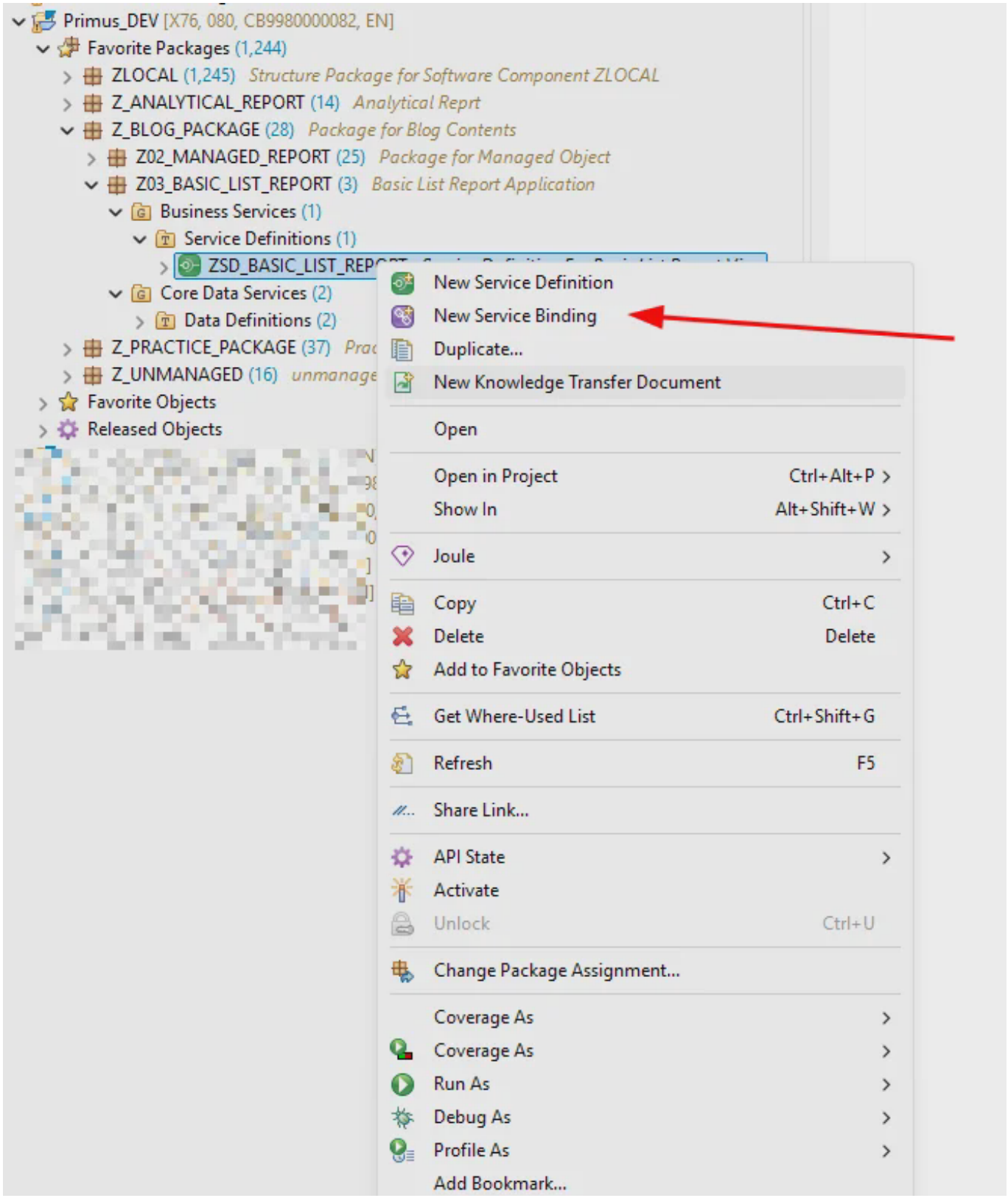
```
@EndUserText.label: 'Service Definition For Basic List Report View'
@ObjectModel.leadingEntity.name: 'ZCDS_C_BILLINGDOC_BASE_CDS'
define service ZSD_BASIC_LIST_REPORT {
  expose ZCDS_C_BILLINGDOC_BASE_CDS as BillingList;
}
```

Activate the Service Definition once complete.

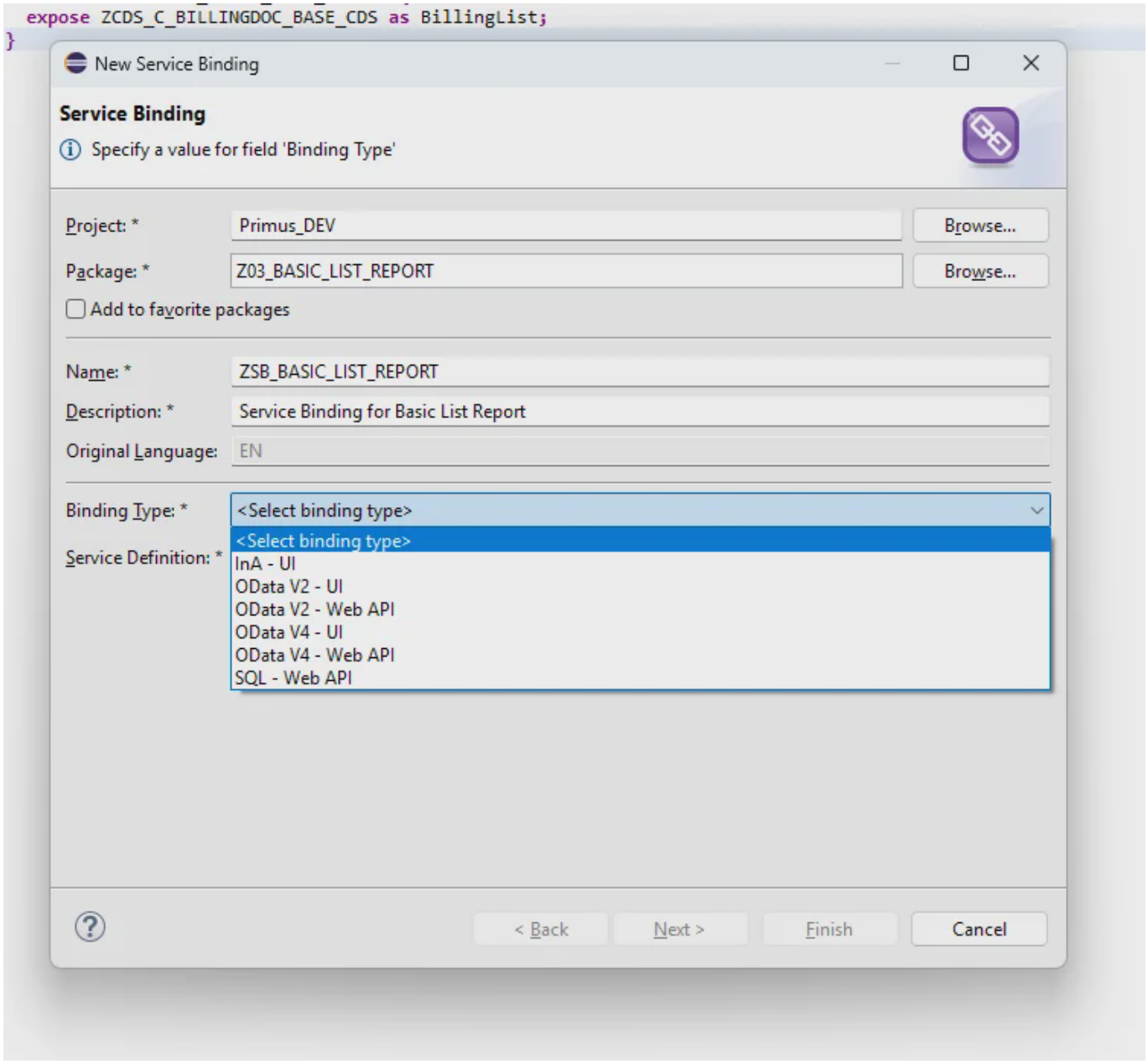


Step 6: Create the Service Binding

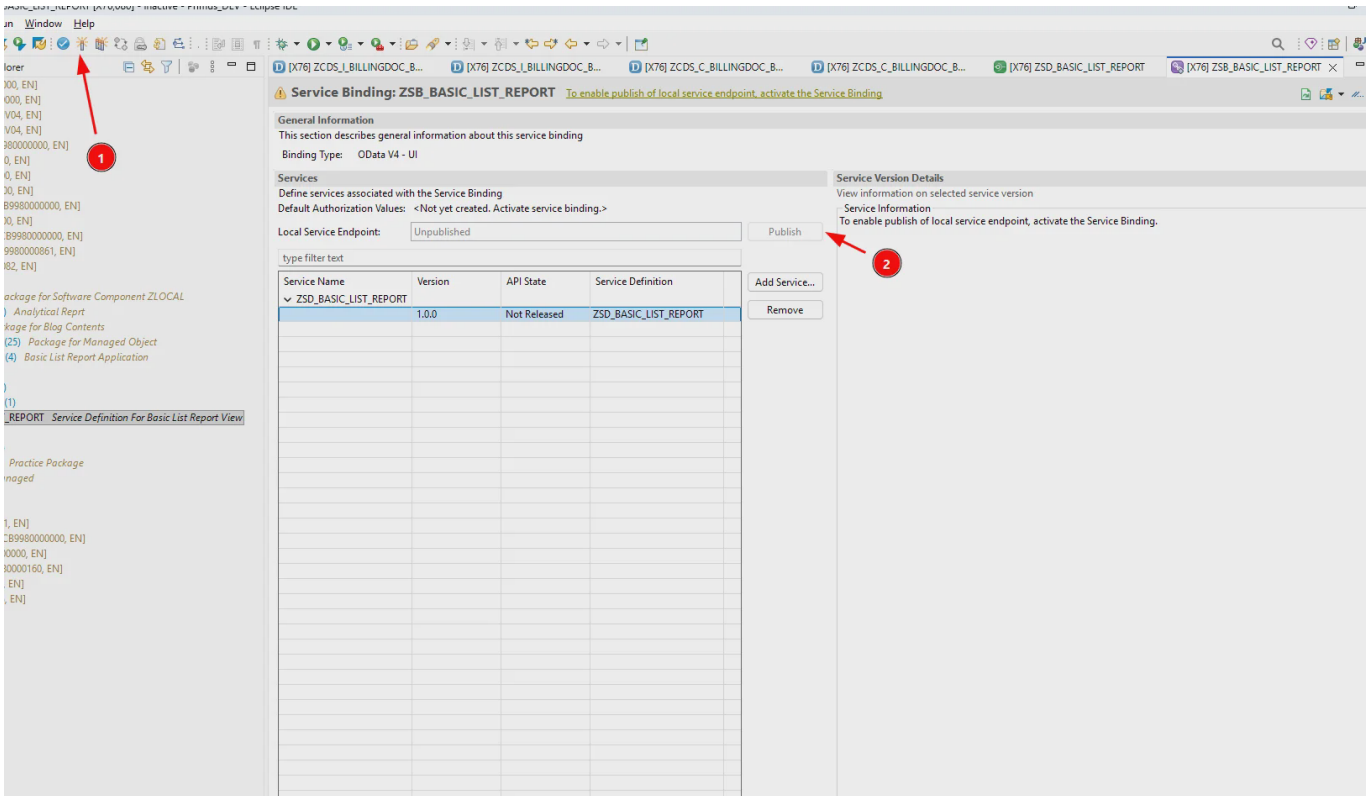
Right-click on the Service Definition and select **New Service Binding**.



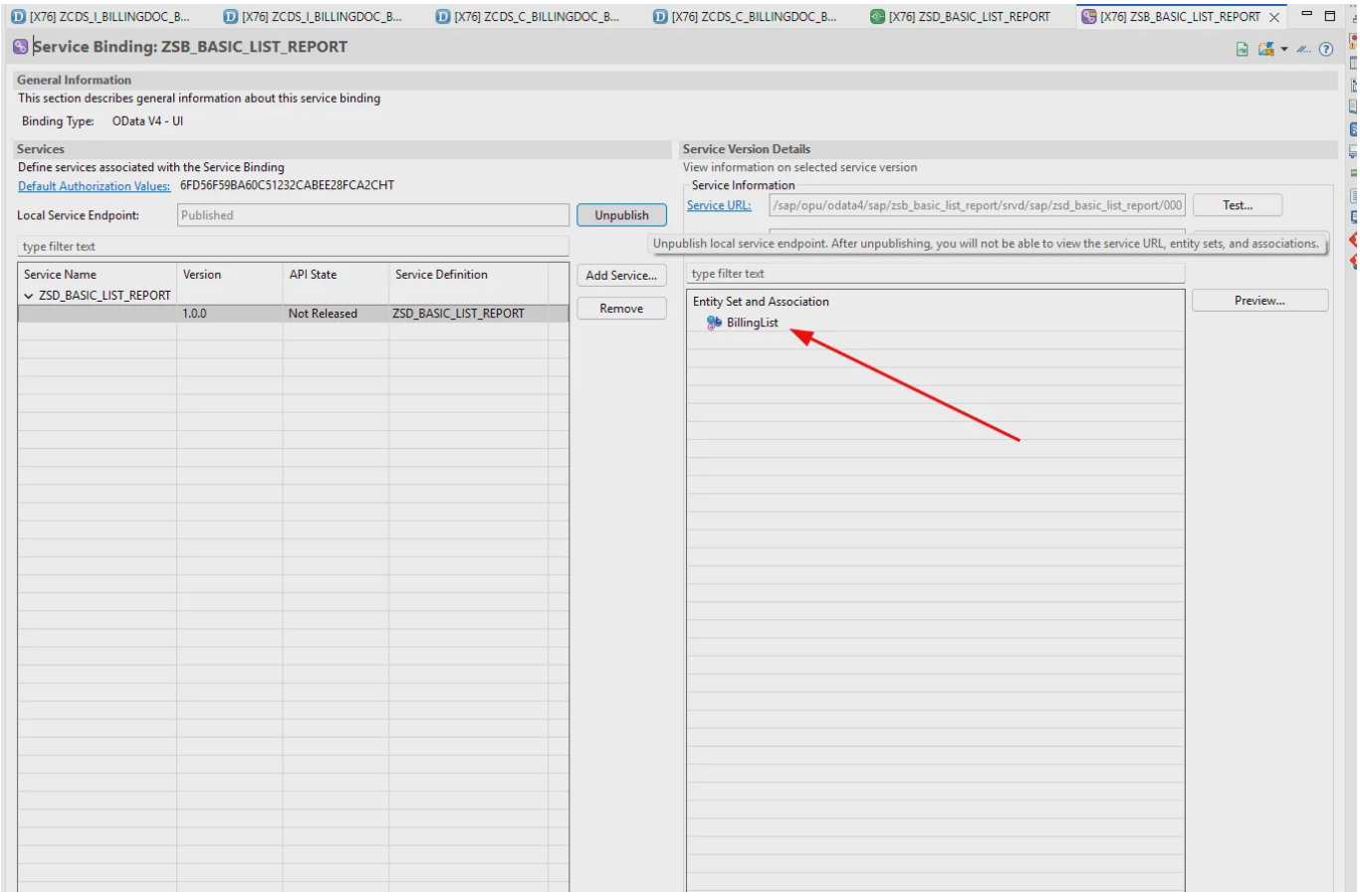
Enter a name and description. Under **Binding Type**, select either **OData V2 - UI** or **OData V4 - UI**. Both are fully supported; this tutorial uses **OData V4 - UI**, as it is the more modern standard and includes performance optimizations and expanded feature support.



On the Service Binding page, first **Activate** the object. This action enables the **Publish** button.

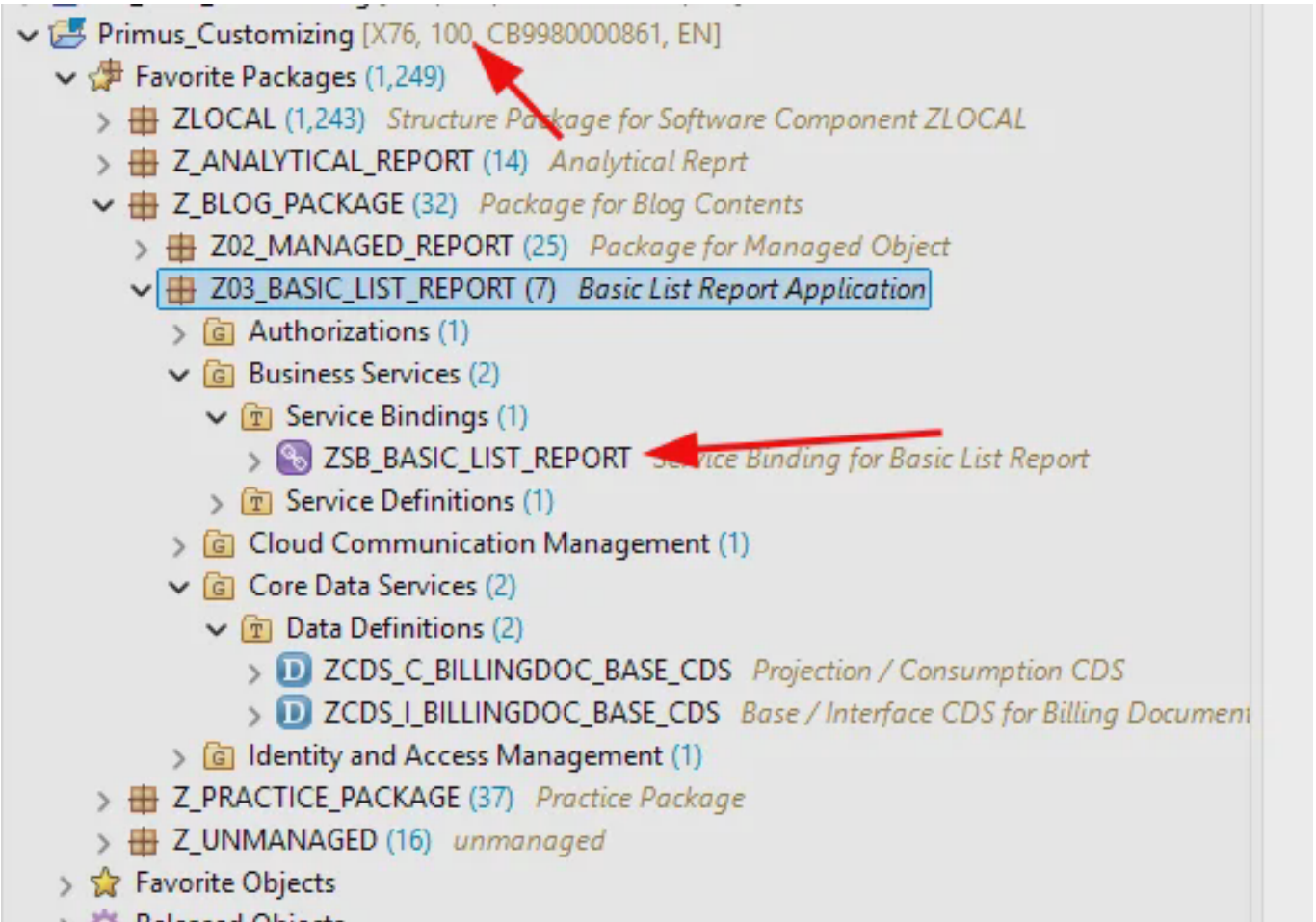


Once published, the entity set defined in the Service Definition will appear in the binding.

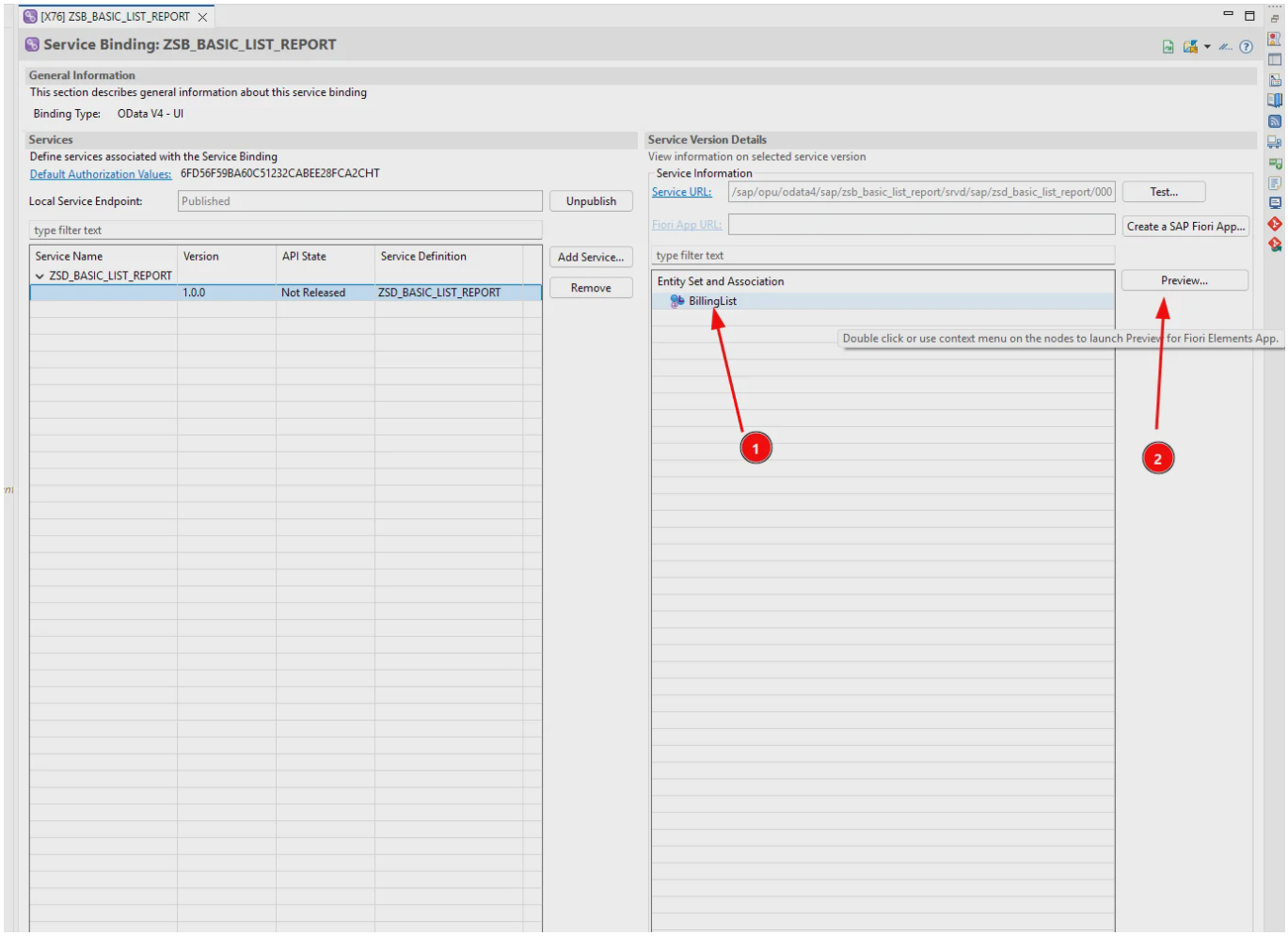


Step 7: Preview the List Report

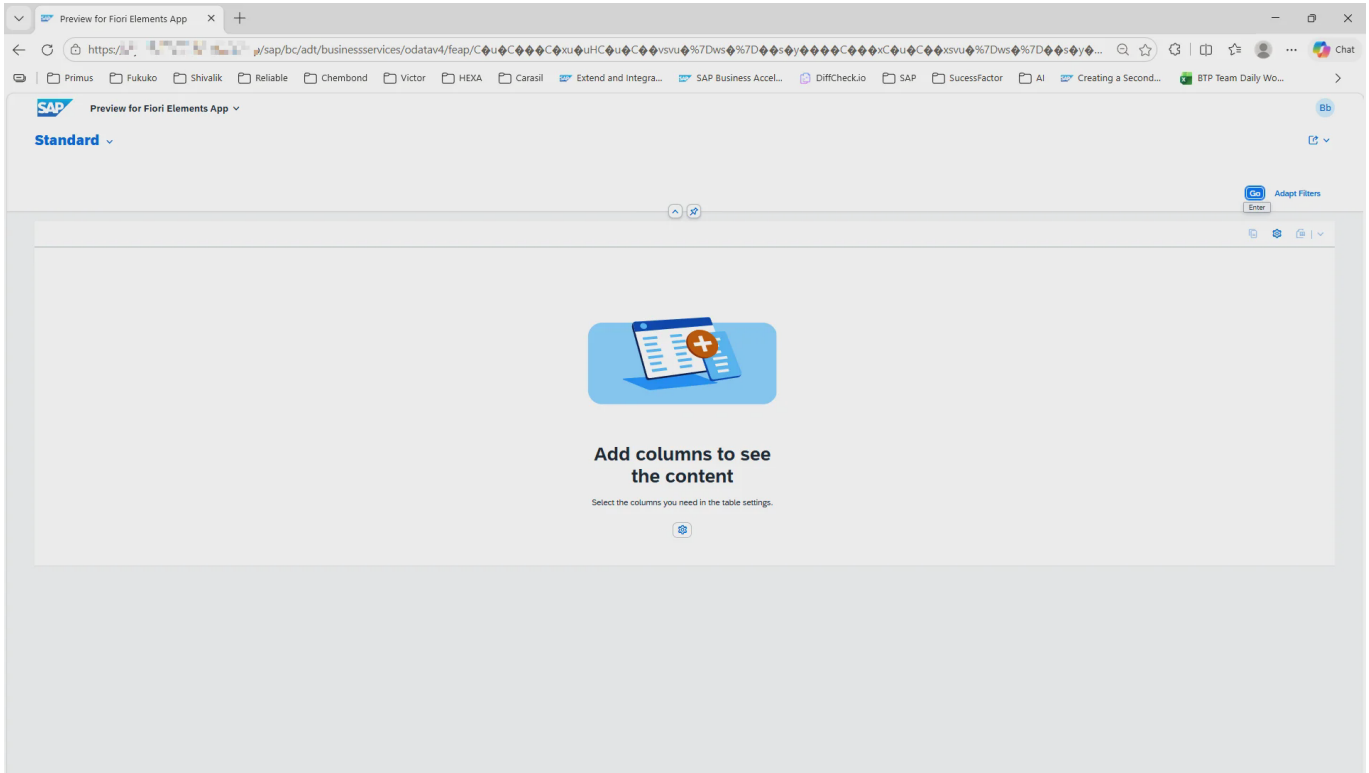
Switch to **Client 100** and open the Service Binding.



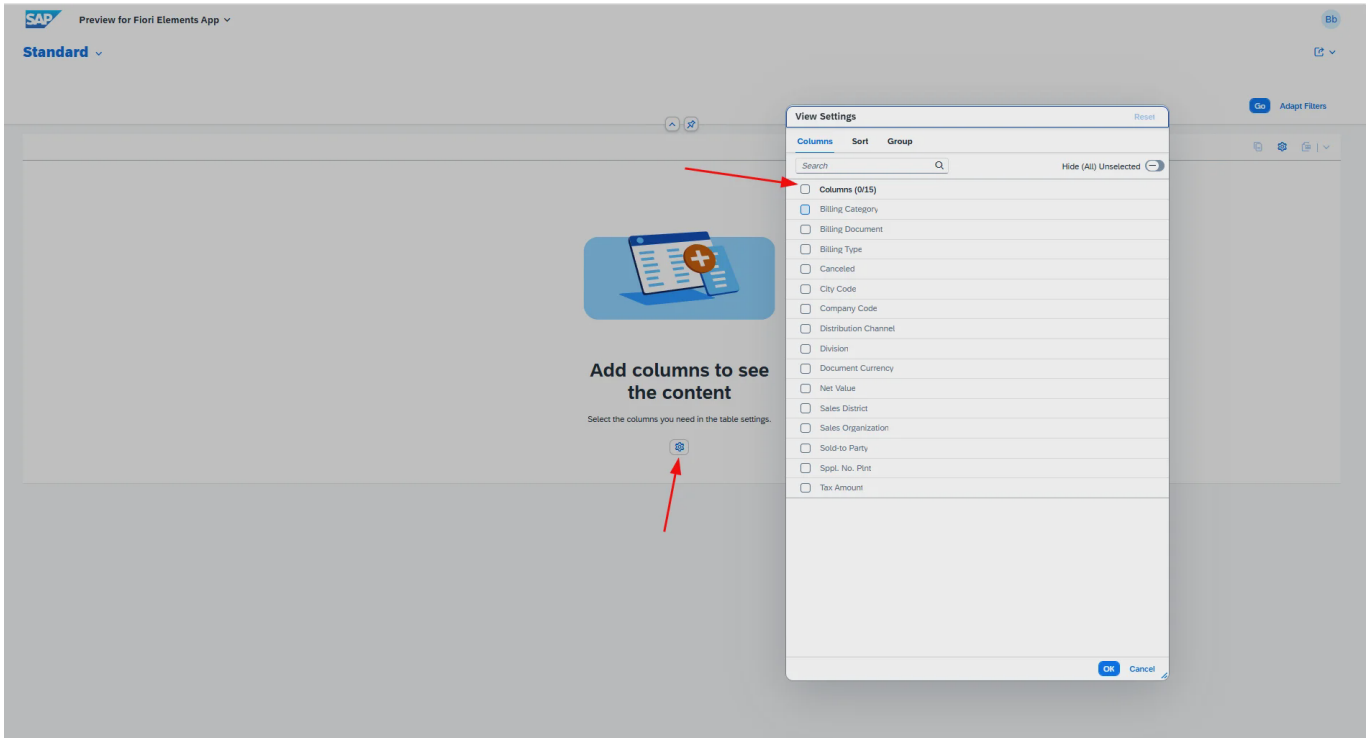
Select the entity set and click **Preview...**.



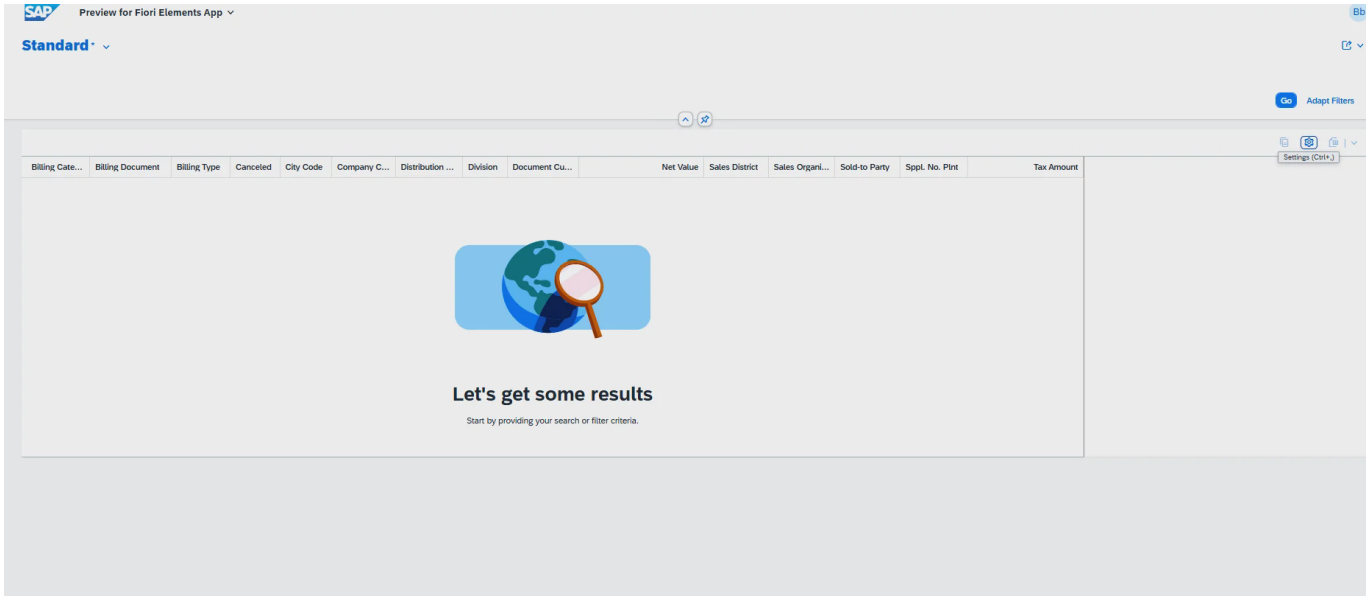
The initial preview will appear with no default columns selected and no visible filter fields -- this is expected behavior at this stage.



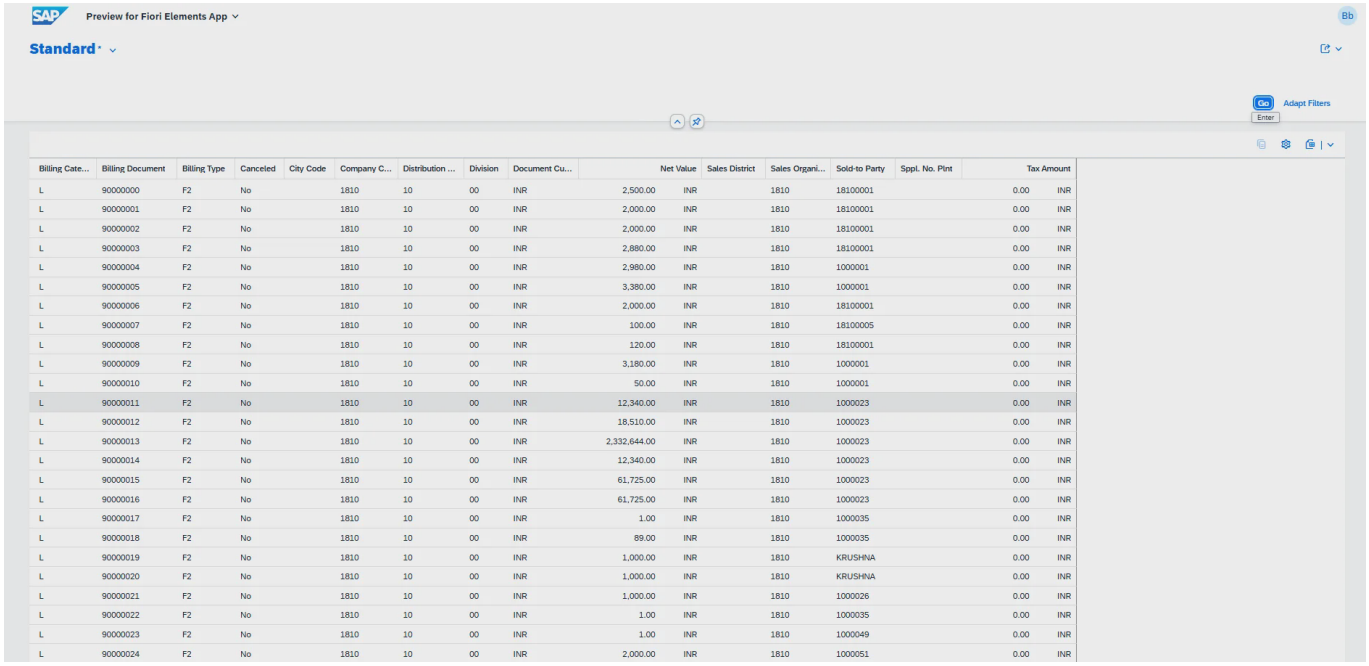
You can manually select columns to display.



After selecting columns, press **Go** to load the data.



The data is now displayed successfully. However, there is a usability gap: columns must be re-selected after every page reload, and no default filter fields are visible. The next steps address both of these issues using CDS annotations.



Billing Cate...	Billing Document	Billing Type	Canceled	City Code	Company C...	Distribution ...	Division	Document Cu...	Net Value	Sales District	Sales Organi...	Sold-to Party	Sppl. No. Plnt	Tax Amount	
L	90000000	F2	No		1810	10	00	INR	2,500.00	INR	1810	18100001		0.00	INR
L	90000001	F2	No		1810	10	00	INR	2,000.00	INR	1810	18100001		0.00	INR
L	90000002	F2	No		1810	10	00	INR	2,000.00	INR	1810	18100001		0.00	INR
L	90000003	F2	No		1810	10	00	INR	2,880.00	INR	1810	18100001		0.00	INR
L	90000004	F2	No		1810	10	00	INR	2,980.00	INR	1810	1000001		0.00	INR
L	90000005	F2	No		1810	10	00	INR	3,380.00	INR	1810	1000001		0.00	INR
L	90000006	F2	No		1810	10	00	INR	2,000.00	INR	1810	18100001		0.00	INR
L	90000007	F2	No		1810	10	00	INR	100.00	INR	1810	18100005		0.00	INR
L	90000008	F2	No		1810	10	00	INR	120.00	INR	1810	18100001		0.00	INR
L	90000009	F2	No		1810	10	00	INR	3,180.00	INR	1810	1000001		0.00	INR
L	90000010	F2	No		1810	10	00	INR	50.00	INR	1810	1000001		0.00	INR
L	90000011	F2	No		1810	10	00	INR	12,340.00	INR	1810	1000023		0.00	INR
L	90000012	F2	No		1810	10	00	INR	18,510.00	INR	1810	1000023		0.00	INR
L	90000013	F2	No		1810	10	00	INR	2,332,644.00	INR	1810	1000023		0.00	INR
L	90000014	F2	No		1810	10	00	INR	12,340.00	INR	1810	1000023		0.00	INR
L	90000015	F2	No		1810	10	00	INR	61,725.00	INR	1810	1000023		0.00	INR
L	90000016	F2	No		1810	10	00	INR	61,725.00	INR	1810	1000023		0.00	INR
L	90000017	F2	No		1810	10	00	INR	1.00	INR	1810	1000035		0.00	INR
L	90000018	F2	No		1810	10	00	INR	89.00	INR	1810	1000035		0.00	INR
L	90000019	F2	No		1810	10	00	INR	1,000.00	INR	1810	KRUSHNA		0.00	INR
L	90000020	F2	No		1810	10	00	INR	1,000.00	INR	1810	KRUSHNA		0.00	INR
L	90000021	F2	No		1810	10	00	INR	1,000.00	INR	1810	1000026		0.00	INR
L	90000022	F2	No		1810	10	00	INR	1.00	INR	1810	1000035		0.00	INR
L	90000023	F2	No		1810	10	00	INR	1.00	INR	1810	1000049		0.00	INR
L	90000024	F2	No		1810	10	00	INR	2,000.00	INR	1810	1000051		0.00	INR

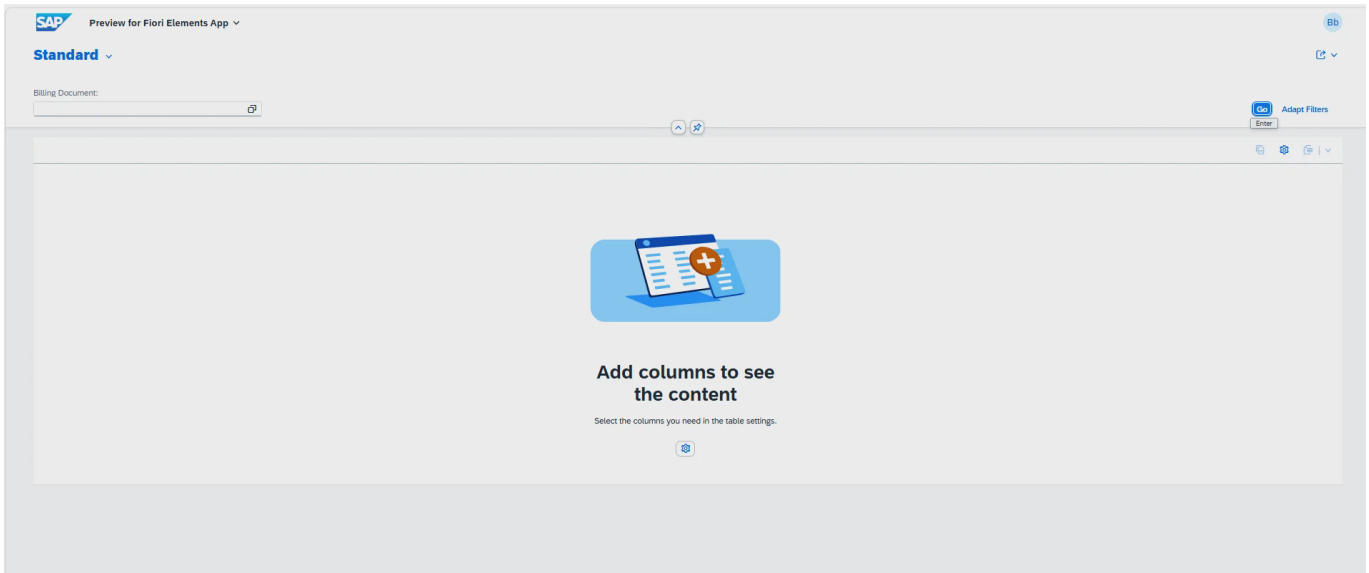
Step 8: Add Default Filters and Columns via Annotations

To define a default filter field, add the following annotation above the relevant field in your Projection View:

```
@UI.selectionField: [{ position: 10 }]
```

To define default visible columns in the list, add:

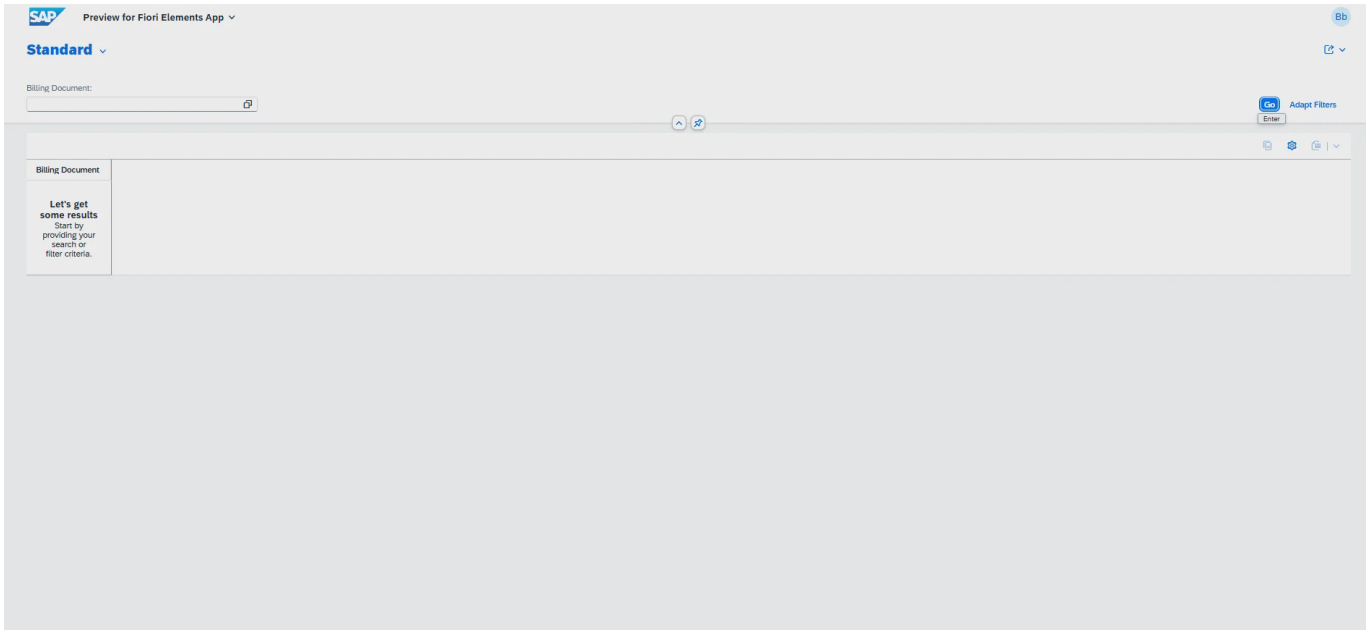
```
@UI.lineItem: [{ position: 10 }]
```



Apply both annotations to the `BillingDocument` field as a starting point.

```
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Projection / Consumption CDS'
@Metadata.ignorePropagatedAnnotations: true
define root view entity ZCDS_C_BILLINGDOC_BASE_CDS
  as projection on ZCDS_I_BILLINGDOC_BASE_CDS
{
      @UI.selectionField: [{ position: 10 }]
      @UI.lineItem: [{ position: 10 }]
  key BillingDocument,
      Division,
      PlantSupplier,
      SoldToParty,
      BillingDocumentCategory,
      BillingDocumentType,
      BillingDocumentIsCancelled,
      SalesDistrict,
      SalesOrganization,
      CityCode,
      DistributionChannel,
      @Semantics.amount.currencyCode: 'TransactionCurrency'
      TotalNetAmount,
      @Semantics.amount.currencyCode: 'TransactionCurrency'
      TotalTaxAmount,
      CompanyCode,
      TransactionCurrency
}
```

On the next reload, the `BillingDocument` column appears by default without any manual selection.

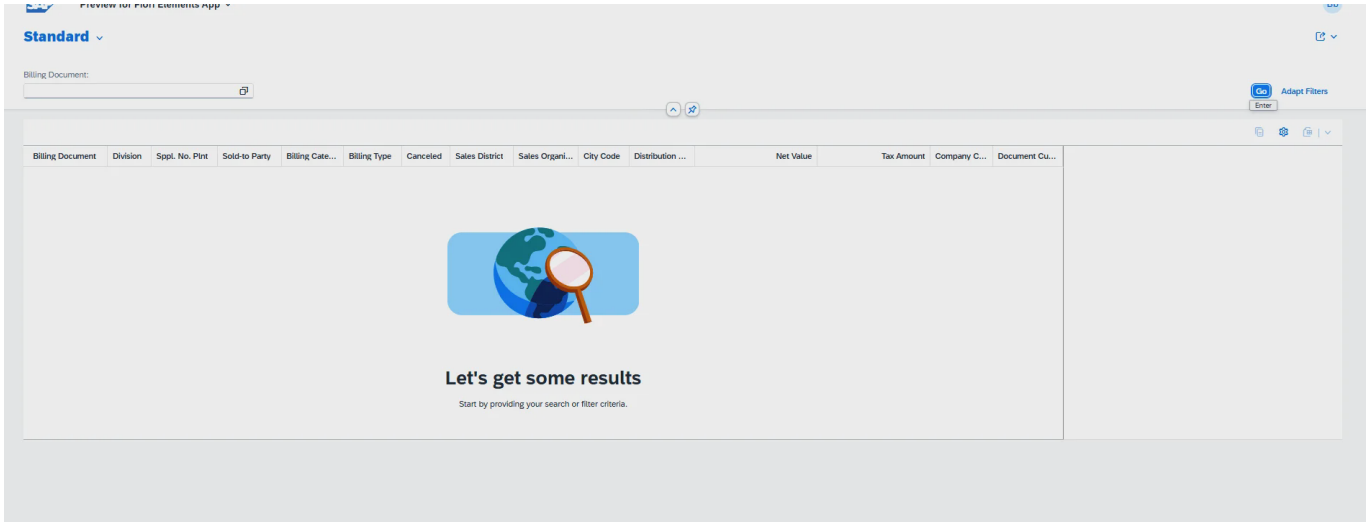


Continue applying both annotations to all remaining fields, adjusting the `position` value to control the display order.

[X76] ZSB_BASIC_LIST_REPORT [X76] ZCDS_C_BILLINGDOC_BASE_CDS

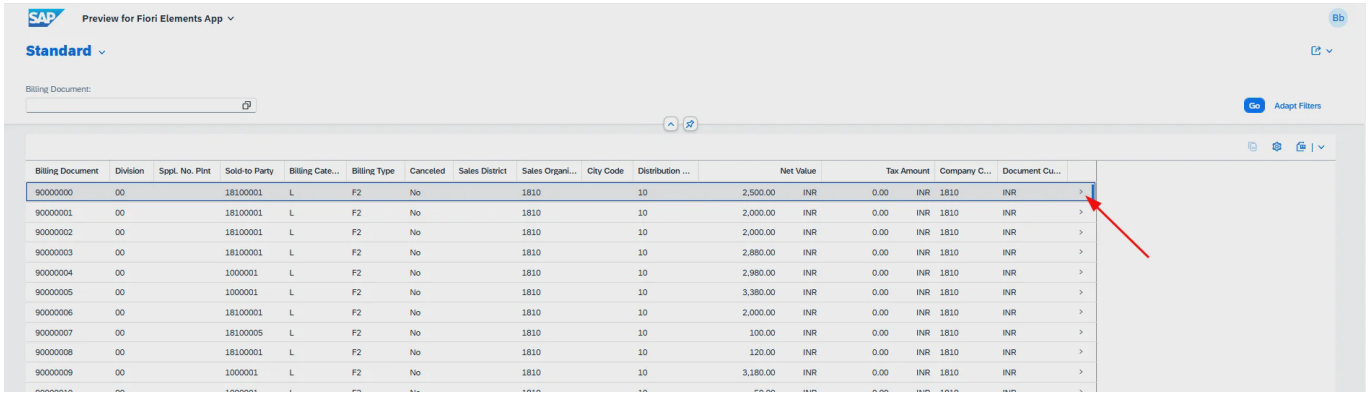
```
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Projection / Consumption CDS'
@Metadata.ignorePropagatedAnnotations: true
define root view entity ZCDS_C_BILLINGDOC_BASE_CDS
  as projection on ZCDS_I_BILLINGDOC_BASE_CDS
{
      @UI.selectionField: [{ position: 10 }]
      @UI.lineItem: [{ position: 10 }]
  key BillingDocument,
      @UI.lineItem: [{ position: 20 }]
      Division,
      @UI.lineItem: [{ position: 30 }]
      PlantSupplier,
      @UI.lineItem: [{ position: 40 }]
      SoldToParty,
      @UI.lineItem: [{ position: 50 }]
      BillingDocumentCategory,
      @UI.lineItem: [{ position: 60 }]
      BillingDocumentType,
      @UI.lineItem: [{ position: 70 }]
      BillingDocumentIsCancelled,
      @UI.lineItem: [{ position: 80 }]
      SalesDistrict,
      @UI.lineItem: [{ position: 90 }]
      SalesOrganization,
      @UI.lineItem: [{ position: 100 }]
      CityCode,
      @UI.lineItem: [{ position: 110 }]
      DistributionChannel,
      @Semantics.amount.currencyCode: 'TransactionCurrency'
      @UI.lineItem: [{ position: 120 }]
      TotalNetAmount,
      @Semantics.amount.currencyCode: 'TransactionCurrency'
      @UI.lineItem: [{ position: 130 }]
      TotalTaxAmount,
      @UI.lineItem: [{ position: 140 }]
      CompanyCode,
      @UI.lineItem: [{ position: 150 }]
      TransactionCurrency
}
```

All configured columns now appear automatically on every page load.

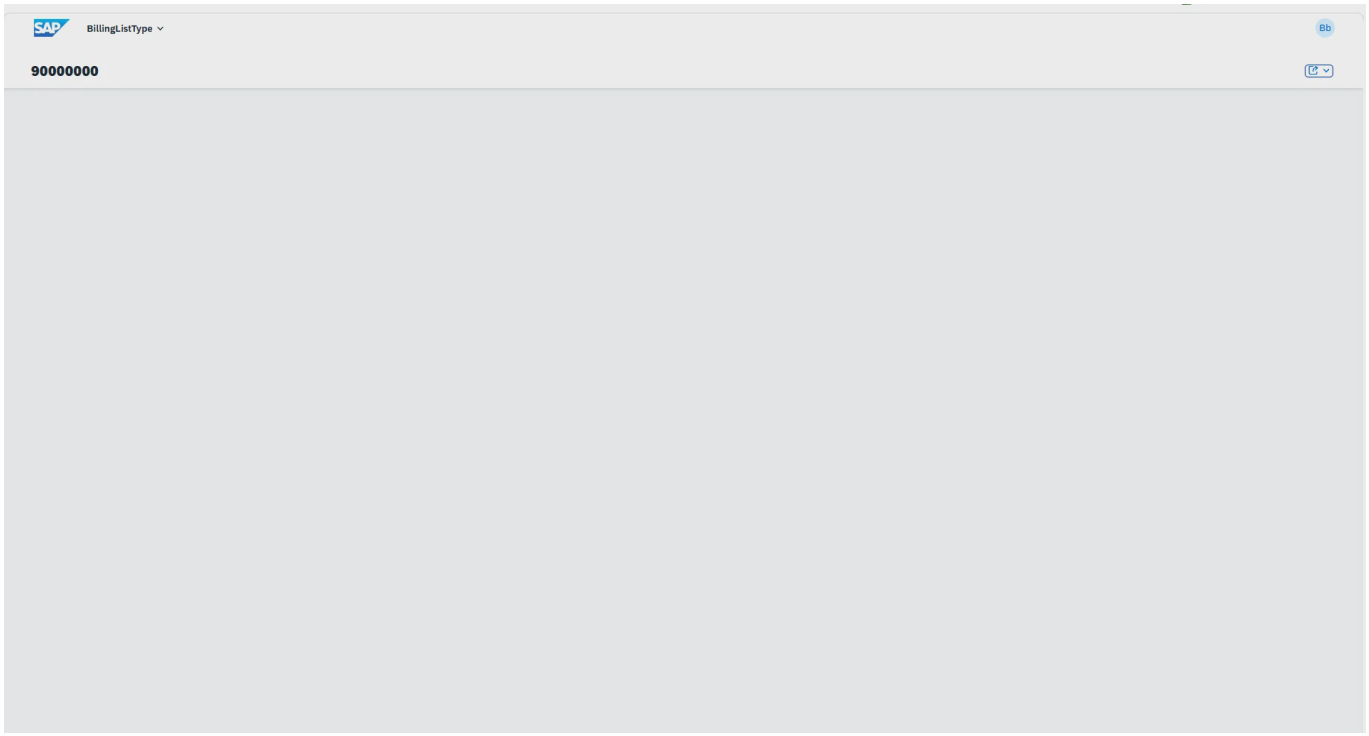


Step 9: Configure the Object Page

Selecting a row in the List Report navigates to the associated **Object Page**. At this point, the Object Page is empty, as no annotations have been defined for it yet.



The page is currently blank.



Start by defining a header for the Object Page. Use the `BillingDocumentType` field as the header title.

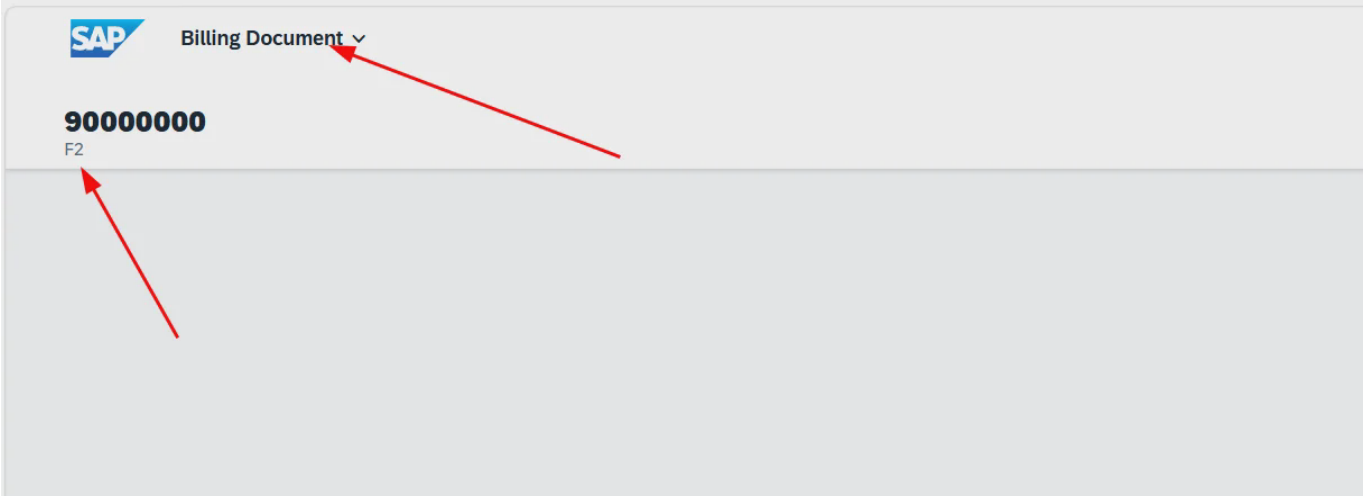
[X76] ZSB_BASIC_LIST_REPORT | [X76] ZCDS_C_BILLINGDOC_BASE_CDS | [X76] Z_C_BILLING_DATA_CDS

```
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Projection / Consumption CDS'
@Metadata.ignorePropagatedAnnotations: true

@UI.headerInfo:{
    typeName: 'Billing Document',
    typeNamePlural: 'Billing Documents',
    title: {
        type: #STANDARD,
        value: 'BillingDocument'
    },
    description: {
        value: 'BillingDocumentType'
    }
}

define root view entity ZCDS_C_BILLINGDOC_BASE_CDS
  as projection on ZCDS_I_BILLINGDOC_BASE_CDS
{
      @UI.selectionField: [{ position: 10 }]
      @UI.lineItem: [{ position: 10 }]
  key BillingDocument,
      @UI.lineItem: [{ position: 20 }]
      Division,
      @UI.lineItem: [{ position: 30 }]
      PlantSupplier,
      @UI.lineItem: [{ position: 40 }]
      SoldToParty,
      @UI.lineItem: [{ position: 50 }]
      BillingDocumentCategory,
      @UI.lineItem: [{ position: 60 }]
      BillingDocumentType,
      @UI.lineItem: [{ position: 70 }]
      BillingDocumentIsCancelled
```

The header now renders correctly on the Object Page.



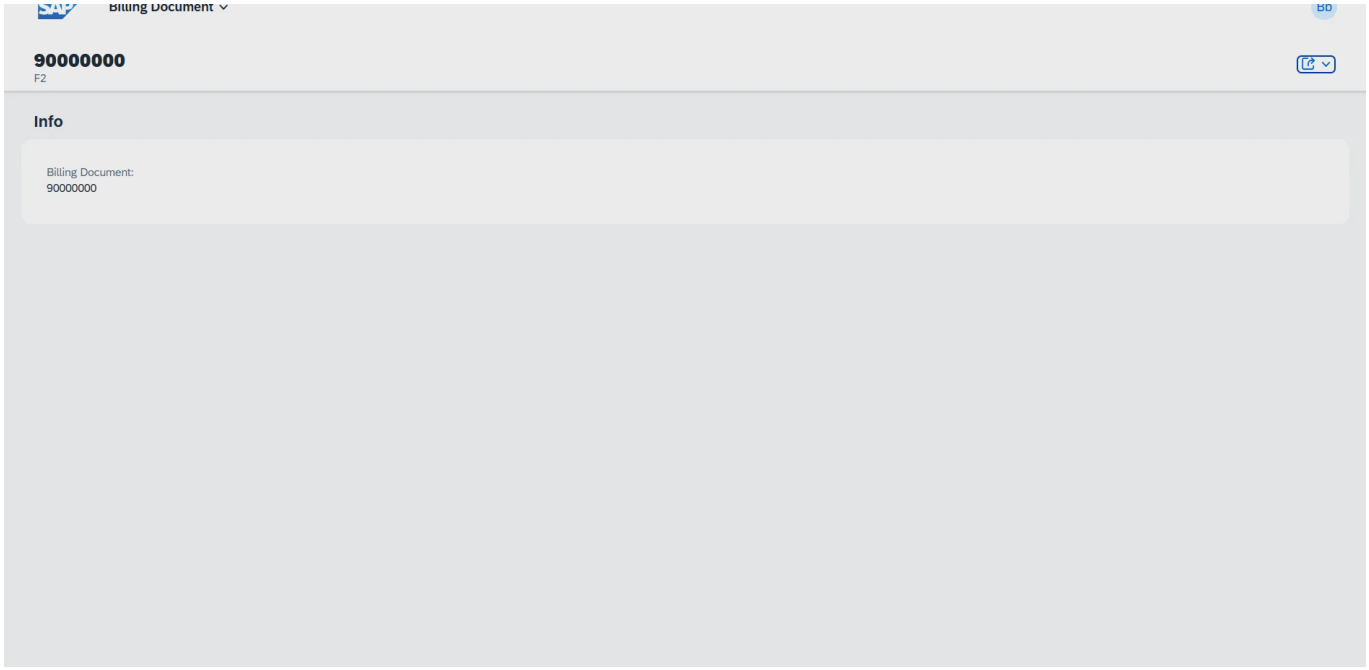
Next, add a field group (panel) to display the detailed record data.

```
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Projection / Consumption CDS'
@Metadata.ignorePropagatedAnnotations: true

@UI.headerInfo:{
    typeName: 'Billing Document',
    typeNamePlural: 'Billing Documents',
    title: {
        type: #STANDARD,
        value: 'BillingDocument'
    },
    description: {
        value: 'BillingDocumentType'
    }
}

define root view entity ZCDS_C_BILLINGDOC_BASE_CDS
  as projection on ZCDS_I_BILLINGDOC_BASE_CDS
{
      @UI.facet: [
                {
                    purpose: #STANDARD ,
                    label: 'Info' ,
                    type: #IDENTIFICATION_REFERENCE,
                    position: 10,
                    id: 'Gninfo'
                }
            ]
      @UI.selectionField: [{ position: 10 }]
      @UI.lineItem: [{ position: 10 }]
      @UI.identification: [{ position: 10 }]
  key BillingDocument,
      @UI.lineItem: [{ position: 20 }]
      Division,
      @UI.lineItem: [{ position: 30 }]
      PlantSupplier,
      @UI.lineItem: [{ position: 40 }]
      SoldToParty,
      @UI.lineItem: [{ position: 50 }]
      BillingDocumentCategory,
      @UI.lineItem: [{ position: 60 }]
      BillingDocumentType,
      @UI.lineItem: [{ position: 70 }]
      BillingDocumentIsCancelled,
      @UI.lineItem: [{ position: 80 }]
      SalesDistrict,
      @UI.lineItem: [{ position: 90 }]
      SalesOrganization,
      @UI.lineItem: [{ position: 100 }]
      CityCode,
      @UI.lineItem: [{ position: 110 }]
      DistributionChannel,
      @Semantics.amount.currencyCode: 'TransactionCurrency'
      @UI.lineItem: [{ position: 120 }]
      TotalNetAmount,
      @Semantics.amount.currencyCode: 'TransactionCurrency'
      @UI.lineItem: [{ position: 130 }]
      TotalTaxAmount,
      @UI.lineItem: [{ position: 140 }]
      CompanyCode,
      @UI.lineItem: [{ position: 150 }]
```

Begin by adding one field to the panel to validate the setup.

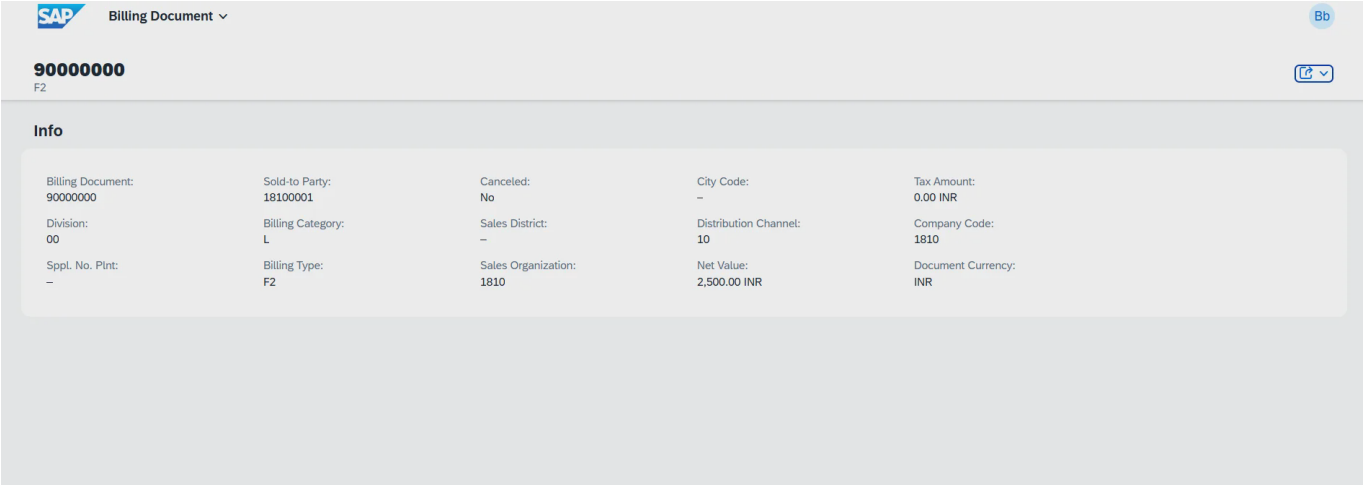


Continue adding the remaining fields to complete the Object Page layout.

```
    description: {
        value: 'BillingDocumentType'
    }
}

define root view entity ZCDS_C_BILLINGDOC_BASE_CDS
  as projection on ZCDS_I_BILLINGDOC_BASE_CDS
{
      @UI.facet: [
                {
                    purpose: #STANDARD ,
                    label: 'Info' ,
                    type: #IDENTIFICATION_REFERENCE,
                    position: 10,
                    id: 'Gninfo'
                }
            ]
      @UI.selectionField: [{ position: 10 }]
      @UI.lineItem: [{ position: 10 }]
      @UI.identification: [{ position: 10 }]
  key BillingDocument,
      @UI.identification: [{ position: 20 }]
      @UI.lineItem: [{ position: 20 }]
      Division,
      @UI.identification: [{ position: 30 }]
      @UI.lineItem: [{ position: 30 }]
      PlantSupplier,
      @UI.identification: [{ position: 40 }]
      @UI.lineItem: [{ position: 40 }]
      SoldToParty,
      @UI.identification: [{ position: 50 }]
      @UI.lineItem: [{ position: 50 }]
      BillingDocumentCategory,
      @UI.identification: [{ position: 60 }]
      @UI.lineItem: [{ position: 60 }]
      BillingDocumentType,
      @UI.identification: [{ position: 70 }]
      @UI.lineItem: [{ position: 70 }]
      BillingDocumentIsCancelled,
      @UI.identification: [{ position: 80 }]
      @UI.lineItem: [{ position: 80 }]
      SalesDistrict,
      @UI.identification: [{ position: 90 }]
      @UI.lineItem: [{ position: 90 }]
      SalesOrganization,
      @UI.identification: [{ position: 100 }]
      @UI.lineItem: [{ position: 100 }]
      CityCode,
      @UI.identification: [{ position: 110 }]
      @UI.lineItem: [{ position: 110 }]
      DistributionChannel,
      @UI.identification: [{ position: 120 }]
      @Semantics.amount.currencyCode: 'TransactionCurrency'
      @UI.lineItem: [{ position: 120 }]
      TotalNetAmount,
      @UI.identification: [{ position: 130 }]
      @Semantics.amount.currencyCode: 'TransactionCurrency'
      @UI.lineItem: [{ position: 130 }]
      TotalTaxAmount,
      @UI.identification: [{ position: 140 }]
```

The Object Page now displays all configured fields in a clean, structured layout.



Summary

In this tutorial, you built a fully functional SAP Fiori Elements List Report using the SAP RAP model in S/4HANA Public Cloud. Here is a recap of everything covered:

1. Created a **Development Package** to organize all RAP artifacts.
2. Built a **Base (Interface) CDS View** using `I_BillingDocument` as the data source, with proper semantic annotations for currency fields.
3. Resolved API contract errors by assigning the appropriate **API State**.
4. Created a **Projection (Consumption) View** to expose the data model to the UI layer.
5. Defined a **Service Definition** to register the projection view as an OData entity.
6. Created and published a **Service Binding** using OData V4.
7. Configured **default filters and columns** using `@UI.selectionField` and `@UI.lineItem` annotations.
8. Enhanced the **Object Page** with a header and field group panels.

This foundation gives you everything you need to extend the application further -- adding actions, associations, draft handling, or custom business logic using the full power of the SAP RAP framework.
