

ABAP Development: Work with Core Data Services (CDS)

## Create a Simple ABAP CDS View in ADT ▼

[Previous](#)[Feedback](#)[Share](#)[Next](#)

# Create a Simple ABAP CDS View in ADT

 Beginner  20 min.  **ABAP Development, Beginner, SAP NetWeaver, Tutorial**

You will learn how to create a CDS (Core Data Services) view using ABAP Development Tools (ADT).

You will learn

- How to use the new Core Data Services (CDS) tools in ABAP Development Tools for Eclipse (ADT).
- How to use the following ABAP and SQL elements in a CDS view:
  - SELECT statement
  - CASE statement
  - WHERE clause



**Julie Plummer** September 18, 2023

Created by  March 9, 2023

Contributors 

- 1 Step 1: [Create a CDS view](#)
- 2 Step 2: [Enter the data source](#)
- 3 Step 3: [Use an existing CDS association](#)
- 4 Step 4: [Add fields from existing associations](#)
- 5 Step 5: [Add a CASE statement](#)
- 6 Step 6: [Add a WHERE clause](#)
- 7 Step 7: [Check your code and view your changes](#)
- 8 Step 8: [Test yourself](#)

[Back to Top](#)

## Prerequisites

- You have a valid instance of an on-premise AS ABAP server, version 7.51 or higher (some ABAP Development Tools may not be available in earlier versions)
- You have run the transaction SEPM\_DG\_OIA\_NEW or transaction STC01 -> tasklist SAP\_BASIS\_EPM\_OIA\_CONFIG. (If you do not, your CDS view will display empty.)
- **Tutorial:** [Create an ABAP Project in ABAP Development Tools \(ADT\)](#)
- **Tutorial:** [Create an ABAP Package](#)

In this tutorial, you will create an ABAP Dictionary-based CDS view. As of ABAP AS 7.57, such views are deprecated. This tutorial is available for compatibility purposes only. For a short, up-to-date tutorial on CDS View Entities, see: **Tutorial:** [Create an ABAP Core Data Services \(CDS\) View in ABAP On-Premise](#)

DS is an extension of the ABAP Dictionary that allows you to define semantically rich data models in the database and to use these data models in your ABAP programs. DS is a central part of enabling code push-down in ABAP applications.

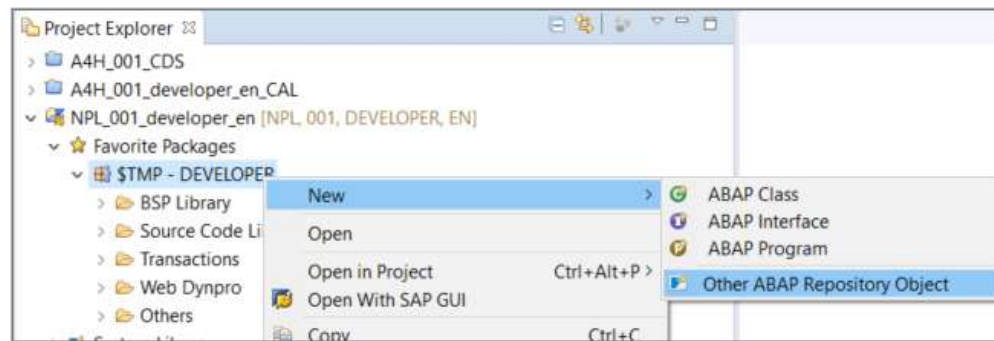
You can find more information about these deprecated CDS Views here:- [ABAP keyword documentation, version 7.51: CDS Views](#)

You can find more information about CDS View Entities here:- [https://help.sap.com/doc/abapdocu\\_latest\\_index\\_html/latest/en-US/index.html?le=abencds\\_v2\\_views.htm](https://help.sap.com/doc/abapdocu_latest_index_html/latest/en-US/index.html?le=abencds_v2_views.htm)- [SAP Community](#). Throughout this tutorial, objects may include the suffix `xxx`. Always replace this with your group number or initials.

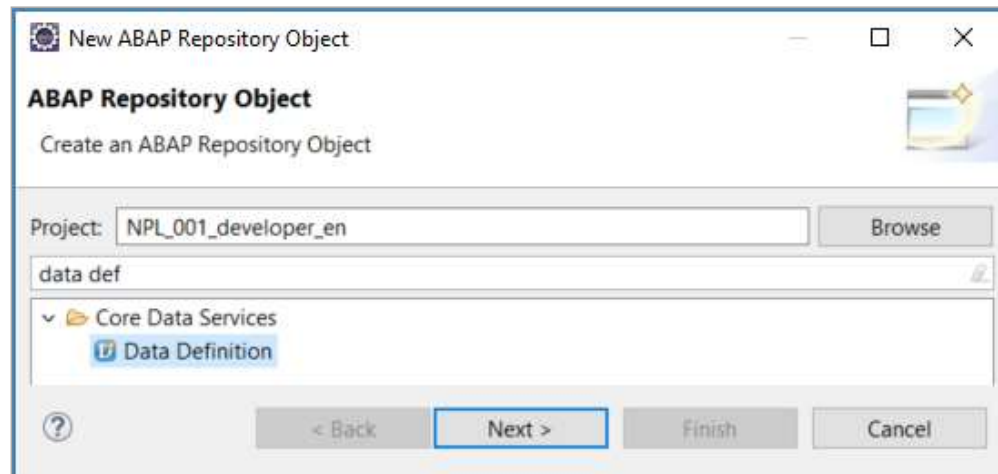
STEP 1

## Create a CDS view

1. In the context menu of your package choose **New** and then choose **Other ABAP Repository Object**.



1. Select **Data Definition**, then choose **Next**.



1. Enter the following values, then choose **Next**:

- Name =
- Description = **Invoice Items**

- Referenced Object: `sepm_sddl_so_invoice_item`

The screenshot shows the 'New Data Definition' dialog box in SAP. The title bar reads 'New Data Definition'. Below the title bar, there is a 'Data Definition' section with the instruction 'Create a data definition'. The dialog contains several input fields and buttons:

- Project:** \*  - Package:** \*  - ☐ Add to favorite packages
- Name:** \*
- Description:** \*
- Original Language:**
- Referenced Object:**

At the bottom of the dialog, there is a help icon (?) and four navigation buttons: '< Back', 'Next >', 'Finish', and 'Cancel'. The 'Next >' button is highlighted with a blue border.

1. Accept the default transport request (local) by simply choosing **Next** again.
2. Select the entry **Define View**, then choose **Finish**

New Data Definition

**Templates**

Select one of the available templates.

☒ Use the selected template

**Define View** Defines a simple projection view with one data source.

**Define View with Join**

**Define View with Association**

**Define View with Parameters**

**Extend View**

```
@AbapCatalog.sqlViewName: '${sql_view_name}'
@AbapCatalog.compiler.compareFilter: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: '${ddl_source_description}'
define view ${ddl_source_name_editable} as select from ${data_source_name}
  ${cursor}
```

[?](#) [< Back](#) [Next >](#) **Finish** [Cancel](#)

[Log in](#) to complete tutorial

Done

STEP 2

Enter the data source

The new view appears in an editor, with the fields from the referenced object, `sepm_sddl_so_invoice_item`. In this editor, enter the following values:

1. Enter `Z_ITEMS_XXX` as the SQL view name.

Your CDS view should now look like this:

```
[UIA] Z_INVOICE_ITEMS_XXX x
1 @AbapCatalog.sqlViewName: 'Z_ITEMS_XXX'
2 @AbapCatalog.compiler.compareFilter: true
3 @AccessControl.authorizationCheck: #NOT_REQUIRED
4 @EndUserText.label: 'Invoice Items'
5
6 define view Z_INVOICE_ITEMS_XXX as select from sepm_sddl_so_invoice_item
7 {
8   key sales_order_invoice_item_key,
9   item_position,
10  sales_order_invoice_key,
11  sales_order_item_key,
12  quantity_unit,
13  quantity,
14  currency_code,
15  gross_amount,
16  net_amount,
17  tax_amount,
18  product_key,|
19  /* Associations */
20  currency,
21  header,
22  product,
23  sales_order_item
24 }
25
```

The SQL view name is the internal/technical name of the view which will be created in the database.

`Z_Invoice_Items` is the name of the CDS view which provides enhanced view-building capabilities in ABAP. You should always use the CDS view name in your ABAP applications.

2. Delete all the fields except:

ABAP key sales\_order\_invoice\_item\_key, currency\_code, gross\_amount

[Log in](#) to complete tutorial

Done

### STEP 3

## Use an existing CDS association

You will now model the relationships between data sources by using some existing CDS associations. You can use associations in path expressions to access elements (fields and associations) in related data sources without specifying JOIN conditions. You can now display the element info by positioning the cursor on the data source name

`sepm_sddl_so_invoice_item` and choosing **F2**.

To see the related data sources that can be accessed using associations, scroll down. To see details about the target data source of the association header, choose the hyperlink `sepm_sddl_so_invoice_header`.



```
define view Z_Invoice_Items as select from sepm_sddl_so_invoice_item {
  sepm_sddl_so_invoice_item.currency_code,
  sepm_sddl_so_invoice_item.gross_amount
}
```

Column	Data Element	Data Type	Description
sales_order_invoice_item_key	snwd_node_key	raw(16)	EPM: Generic Node Key
item_position	snwd_so_item_pos	char(10)	EPM: Sales Order Item Position
sales_order_invoice_key	snwd_node_key	raw(16)	EPM: Generic Node Key
sales_order_item_key	snwd_node_key	raw(16)	EPM: Generic Node Key
quantity_unit	snwd_quantity_unit	unit(3)	EPM: Quantity Unit
quantity	snwd_quantity	quan(13,3)	EPM: Quantity
currency_code	snwd_curr_code	cuky(5)	EPM: Currency Code
gross_amount	snwd_ttl_gross_amount	curr(15,2)	EPM: Total Gross Amount
net_amount	snwd_ttl_net_amount	curr(15,2)	EPM: Total Net Amount
tax_amount	snwd_ttl_tax_amount	curr(15,2)	EPM: Total Tax Amount
product_key	snwd_node_key	raw(16)	EPM: Generic Node Key

Association	Target	Cardinality
currency	tcure	0..1
header	sepm_sddl_so_invoice_header	0..1
product	sepm_sddl_product	0..1
sales_order_item	sepm_sddl_salesorder_item	0..1

[Log in](#) to complete tutorial

Done

## STEP 4

### Add fields from existing associations

You will now add fields of related data sources to the SELECT list of `Z_Invoice_Items`, using the associations in path expressions. Each element in the path expression must be separated by a period.

1. Add the association `header` to your selection list, preferably with a comment, by adding the following the code. do not forget to add a comma after the previous item, `gross_amount`:

ABAP

[Copy](#)

```

1 | //      * Associations *//
2 |      header
3 |

```

```

19 |
20 | /* Associations */
21 | header
22 | }
23 |

```

2. You will get an error, "Field header must be included in the selection list together with field `SEPM_SDDL_SO_INVOICE_ITEM.SALES_ORDER_INVOICE_KEY`". Resolve this by adding the field `sepm_sddl_so_invoice_item.sales_order_invoice_key` to the Select statement.
3. Add the `company_name` of the business partner to the SELECT list using the associations **header** and **buyer** in a path expression
4. Add the `payment_status` from the invoice header to the SELECT list using the association **header**

ABAP

[Copy](#)

```

1 | header.payment_status

```

```
define view Z_Invoice_Items_3
as select from sepm_sddl_so_invoice_item

{
  header.buyer.company_name,
  sepm_sddl_so_invoice_item.sales_order_invoice_key,
  sepm_sddl_so_invoice_item.currency_code,
  sepm_sddl_so_invoice_item.gross_amount,
  header.payment_status,

  /* Associations */
  header
}
}
```

[Log in](#) to complete tutorial

Done

## STEP 5

### Add a CASE statement

If the invoice has been paid, you want to set the `payment_status` to X (true). Do this by implementing a CASE expression, assigning the alias `payment_status` to the CASE expression.

Remove the existing declaration, `header.payment_status,` and replace it with the following code. Do not forget to separate the new, calculated field `paid` and the association `header` with a comma.

ABAP

[Copy](#)

```
1 case header.payment_status
2   when 'P' then 'X'
3   else ''
4 end as paid,
5
```

```
18 case header.payment_status
19   when 'P' then 'X'
20   else ''
21 end as paid,
22
23 //      * Associations *//
24 header
25 }
```

You can check your code below.

[Log in](#) to complete tutorial

Done

STEP 6

## Add a WHERE clause

You will now filter the results so that only invoice items with

`currency_code = 'EUR'` are retrieved.

1. Add a WHERE clause:

ABAP

[Copy](#)

```
1 | WHERE currency_code = 'EUR'  
2 |
```

```
25 }  
26  
27 where  
28   currency_code = 'EUR'
```

2. Save and activate the data definition by choosing **Save** (`Ctrl+S`) and **Activate** (`Ctrl+F3`).



[Log in](#) to complete tutorial

Done

Check your code and view your changes

STEP 8

## Test yourself

Which of the following is not a valid association path? Choose one answer only.

- ☐ sepm\_sddl\_so\_invoice\_item.product.changed\_by
- ☐ sepm\_sddl\_so\_invoice\_item.product.dim\_unit
- ☐ sepm\_sddl\_so\_invoice\_item.product.supplier
- ☐ sepm\_sddl\_so\_invoice\_item.product.storage\_bins

[Log in](#) to complete tutorial

[Check answer](#)

ext Steps

TUTORIAL



## Display a CDS View Using ALV with IDA

 15 min.

### 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](#)

[Cookie Preferences](#)

---

Find us on

