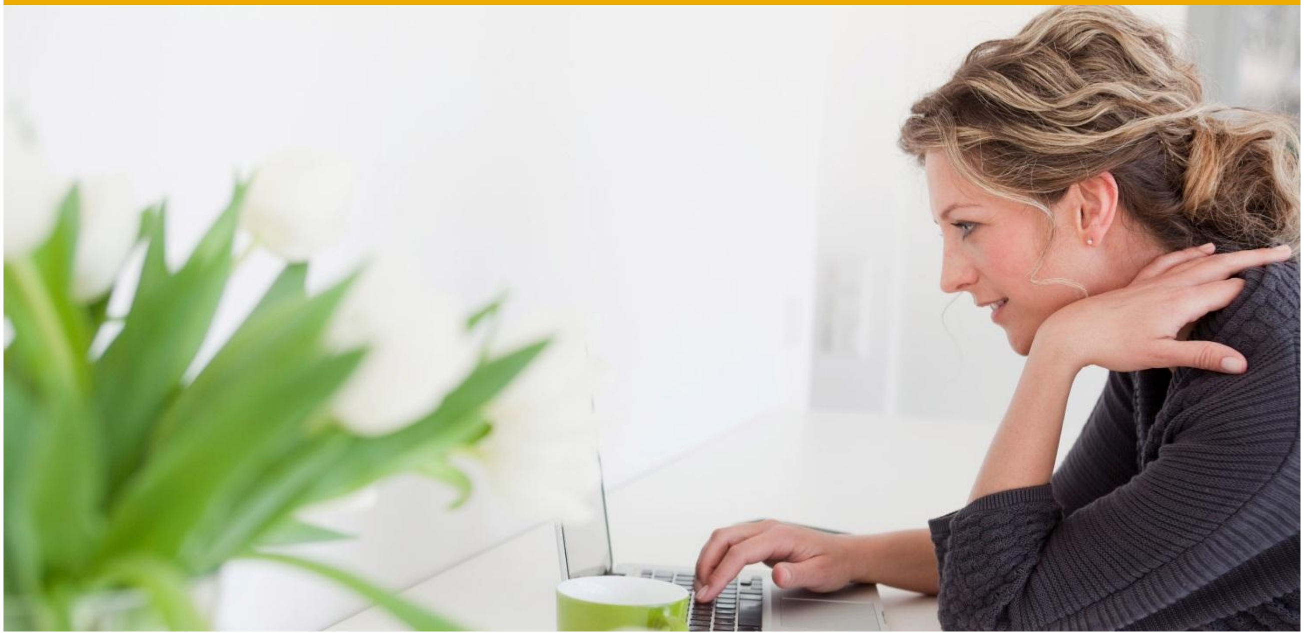


How to Implement a HANA View and Access it Natively in ABAP Using External View and Open SQL

Step-by-Step Tutorial

Release date: July 19, 2013

Table of Contents

Prerequisites **3**

About SAP HANA Attribute Views **3**

Tutorial Objectives **3**

Use Case Description **3**

Procedure Overview **4**

Step-by-Step Procedure **5**

Step 1: Implement a SAP HANA Attribute View 5

Step 2: Expose a HANA View as External View in the ABAP Dictionary 12

Step 3: Implement an ABAP Report consuming an External View 14

Summary **16**

Related Content **16**

This tutorial demonstrates how to use the new AS ABAP built-in possibilities provided with SAP NetWeaver 7.4 to leverage the power of SAP HANA - especially of SAP HANA views - in your ABAP applications.

You will learn how to create a SAP HANA *Attribute View*, expose it in the application server ABAP using the new DDIC entity called *External View* and then access it natively in an ABAP program (consumption in ABAP).

For more information and guides about the development of *ABAP for SAP HANA* applications - meaning applications built out of ABAP and SAP HANA development objects-, visit our SCN Page http://scn.sap.com/docs/DOC-35518.

Prerequisites

- SAP NetWeaver AS ABAP 7.4 SP2 (or higher) running on SAP HANA
- SAP HANA Appliance Software SPS 04 (or higher)
- SAP HANA DB SQLScript V2.0 (or higher)
- SAP HANA Studio
- ABAP Development Tools for SAP NetWeaver

About SAP HANA Attribute Views

SAP HANA provides three types of HANA views: *Attribute View*, *Analytic View* and *Calculation View*.

Attribute Views are used to model an entity based on the relationships between attribute data that are located in one or multiple HANA database tables. They define joins between tables and select a subset of their columns. An attribute view is primarily used as dimensions of analytic views and search models

More information about the different HANA views is available in the SAP HANA Studio Help (menu entry *Help > Help Content)* and in the SAP HANA Developer Guide.

Tutorial Objectives

After completing this tutorial, you will be able to:

- Implement a SAP HANA Attribute View
- Expose a HANA view in the AS ABAP using the new DDIC entity called *External View*
- Implement the consumption of a HANA view in ABAP using *Open SQL* (native access)

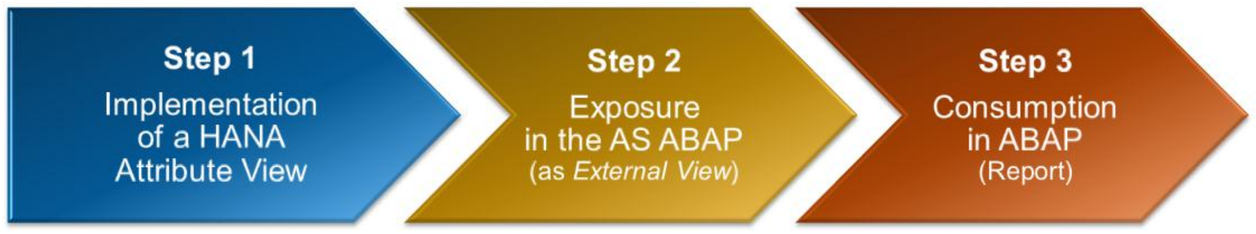
Use Case Description

The Account Receivables accountant of your company wants to have a look at the customers open sales order invoice items. The number of open days per sales order and some customer (aka business partner) data of the assigned buyer should be displayed.

Around 1 million sales order (corresponding to more or less 6 million sales order invoice) will be processed on-the-fly.

More information about the *Open Items Analytics* reference scenario underlying this use case is available under http://scn.sap.com/docs/DOC-41248.

Procedure Overview



* A HANA *Attribute View* will be created in Step 1, but note that the two following steps (1 and 2) are identical for a HANA *Calculation View* and a *HANA Analytical View*.

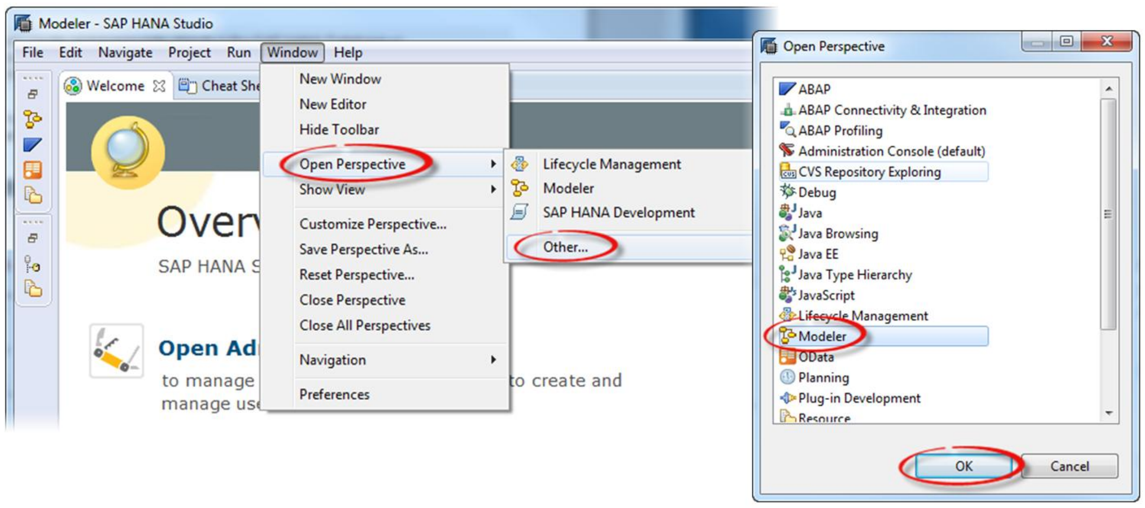
Step-by-Step Procedure

Step 1: Implement a SAP HANA Attribute View

You will create a HANA *Attribute View* which contains the number of open days per sales order and some customer (business partner) data of the assigned buyer. The number of open day per sale order will be calculated on-the-fly in the view.

Info: You need a database user in order to create and manipulate objects in the SAP HANA Database.
It is recommended to create your local test objects in package *system-local.private* located under the *Content* folder of your database system in *Modeler* perspective.

1. Start the *SAP HANA Studio* and open the *Modeler* perspective.
   Select menu entry *Window* > *Open perspective* > *Others…*, choose Modeler from the appearing dialog box and press *OK*.



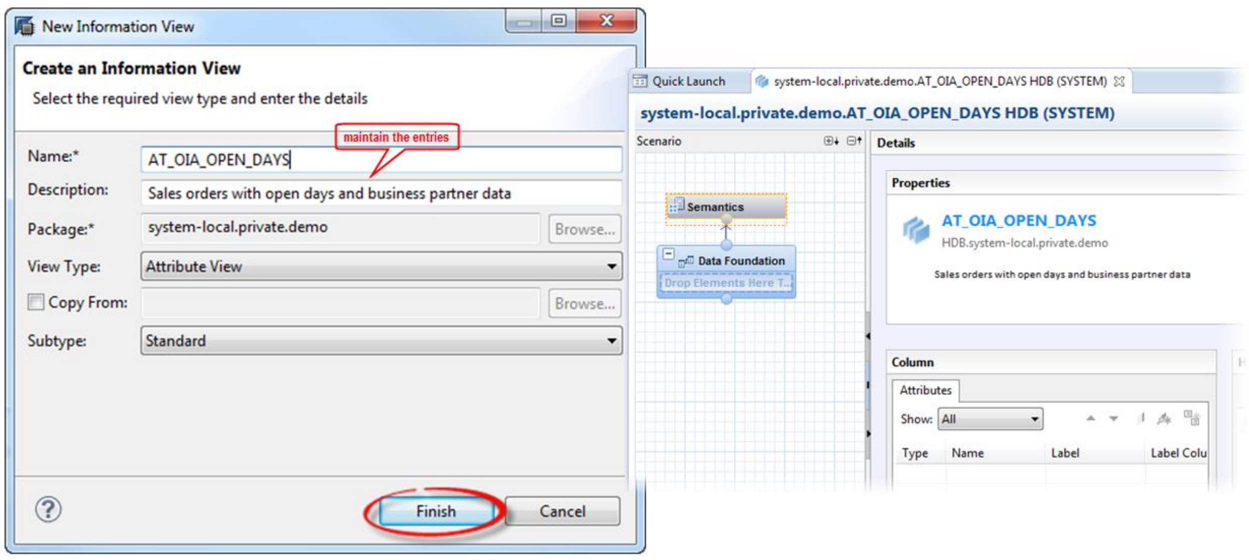
2. Go to the *SAP HANA Systems* navigator view, select the relevant connection and navigate to the package of the *Content* folder where the new view should be created.

   How to add a system connection to the *Modeler* perspective of the SAP HANA Studio:
   Right-click the *SAP HANA Systems* navigator view, select context menu entry *Add System…*, maintain the required information (*Hostname*, *Instance Number*, etc.) in the appearing dialog and confirm.
   More detailed information on this can be found under menu *Help* > *Help Contents* of the SAP HANA Studio.

3. *Optional*: If not yet available, create a new package (e.g. *system-local.private.**demo***) for your test objects.

   Note: It is recommended to create local test objects (e.g. packages and HANA entities) in the package *system-local.private* located under the *Content* folder of your SAP HANA system.

4. Create a new *Attribute View named AT_OIA_OPEN_DAYS*.
   Right-click your package (e.g. *system-local.private.demo)*, select context menu entry *New > Attribute View…, m*aintain the required entries in the creation wizard (see below) and press on *Finish*.
   - *Name* : *AT_OIA_OPEN_DAYS*
   - Description : *Sales orders with open days and business partner data*
   - *Default Schema: ABAP (SAP<SID>)*
   - *Run With* : *Invoker's Rights*



New Information View

Create an Information View

Select the required view type and enter the details

Name:* AT_OIA_OPEN_DAYS

Description: Sales orders with open days and business partner data

Package:* system-local.private.demo

View Type: Attribute View

Copy From:

Subtype: Standard

system-local.private.demo.AT_OIA_OPEN_DAYS HDB (SYSTEM)

AT_OIA_OPEN_DAYS

HDB.system-local.private.demo

Sales orders with open days and business partner data

Show: All

> **Information about the *SAP<SID>*:**
> An application server ABAP is installed in the database in a schema, the so-called *SAP<SID>*, so you can work in the database with multiple schemas.
>
> For example, the schema *SAPABC* belongs to the system whose SID is *ABC*.

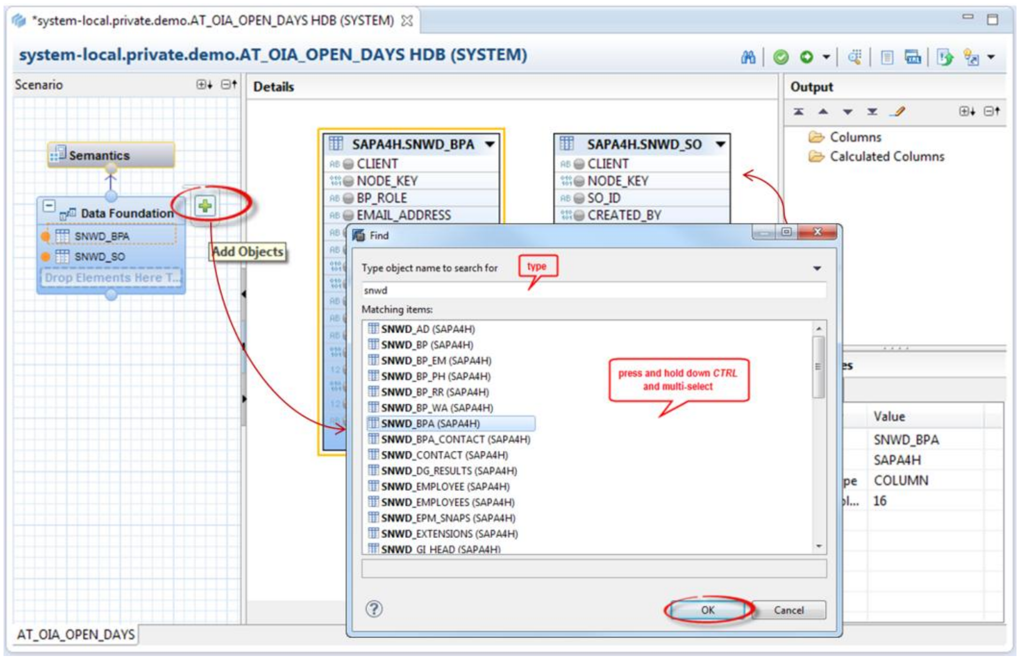
More information about the different properties can be found under *Help > Help Contents*.

5. Add the two tables *SAP<SID>.SNWD_SO* (containing information about sales orders) and *SAP<SID>.SNWD_BPA* (containing business partner master data) which are required for building the view.
   Go to the editor of the view, select (and hover) the *Data Foundation* node in the *Scenario* panel and click on the appearing *Add Objects* icon (✚). Search for '*SNWD'* in the *Find* dialog. Select table *SAP<SID>.SNWD_SO* and press *OK*.

   Repeat the same steps for adding table *SAP<SID>.SNWD_BPA* to the view.

   You can add both tables in one step by pressing and holding down the *Ctrl* key, then multi-selecting the entries and pressing *OK*.

> **Note:** The tables *SAP<SID>.SNWD_SO* and *SAP<SID>.SNWD_BPA* will be mentioned without their prefix (*SAP<SID>.*) in the rest of this tutorial. Meaning, they will be simply mentioned as *SNWD_SO* and *SNWD_BPA*.

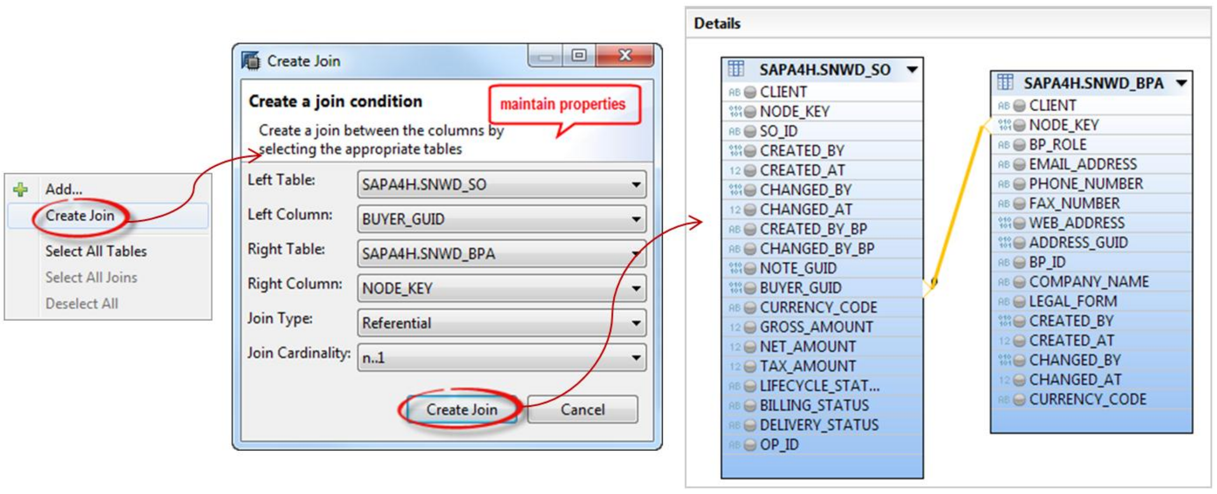


In the *Attribute View*, both tables are now graphically displayed in the *Details* panel of the *Data Foundation* node.

6. Create a *Referential Join* condition in order to combine a sales order with the assigned buyer. Right-click the *Details* panel content area, select context menu entry *Create Join,* maintain the required entries (see below) and press the *Create Join* button.

    - **Left** Table : *SNWD_SO*
    - **Left** Column : *BUYER_GUID*
    - **Right** Table : *SNWD_BPA*
    - **Right** Column : *NODE_KEY*
    - Join Type : *Referential*
    - Join Cardinality : *n..1*

    The *referential join* dependency is displayed with a line between the two tables - especially between table columns *SNWD_SO.BUYER_GUID* and *SNWD_BPA.NODE_KEY.*



Left Table: SAPA4H.SNWD_SO
Left Column: BUYER_GUID
Right Table: SAPA4H.SNWD_BPA
Right Column: NODE_KEY
Join Type: Referential
Join Cardinality: n..1

After creating a join, you can edit its properties in the *Properties* view.

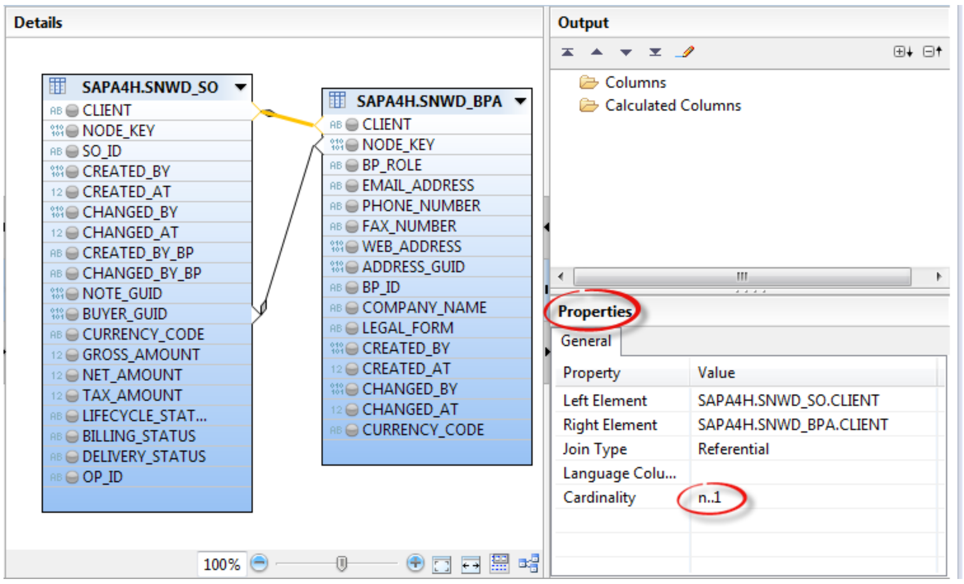
Note about Joins:
Joins are used to query data from more than one table. 4 join types can be created in the SAP HANA Studio: *Referential*, *Inner*, *Left Outer*, *Right Outer* and *Text Join*.

- A *Inner Join* returns rows when there is at least one match in both sides of the join.
- A *Referential Join* is semantically an inner join that assumes that referential integrity is given which means that the left table always has a corresponding entry in the right table.
- A *Text Join* between table fields is used to get language-specific data.

More information about the different *Join* types can be found in the *Help*.

7. Create a *Referential* join condition in order to combine data from the same system client.
   Let's create the new join graphically (instead of using context menu entry *Create Join*) by drawing a line between the CLIENT columns of both tables. For that, just left-click and hold down *SNWD_SO.CLIENT* and draw the line onto column *SNWD_BPA.CLIENT*.

   Afterwards the join conditions between both tables will look like in the picture below. Make sure the cardinality is set to *n..1*.



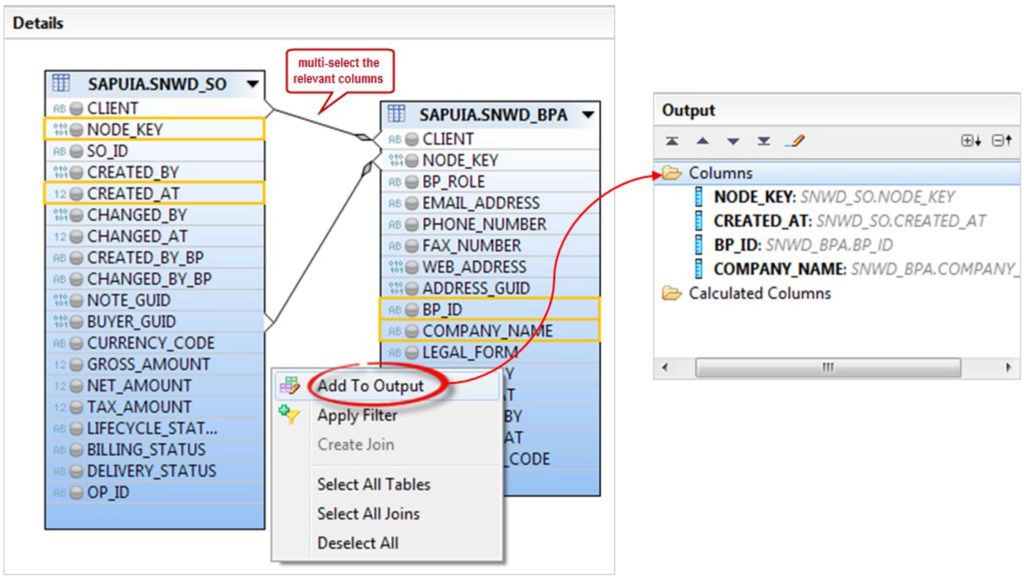
Property	Value
Left Element	SAPA4H.SNWD_SO.CLIENT
Right Element	SAPA4H.SNWD_BPA.CLIENT
Join Type	Referential
Language Colu...	
Cardinality	n..1

8. Save the current state of the view using the appropriate toolbar icon (or *Ctrl+S*) and go ahead with the implementation.

9. Add columns *NODE_KEY and CREATED_AT* of table *SNWD_SO* and columns *BP_ID* and *COMPANY_NAME* of table *SNWD_BPA* to the *Output Columns* of the view.
   Press and hold down the *Ctrl* key and multi-select the relevant table columns: *SNWD_SO.NODE_KEY, SNWD_SO.CREATED_AT*, *SNWD_BPA.BP_ID* and *SNWD_BPA.COMPANY_NAME*.

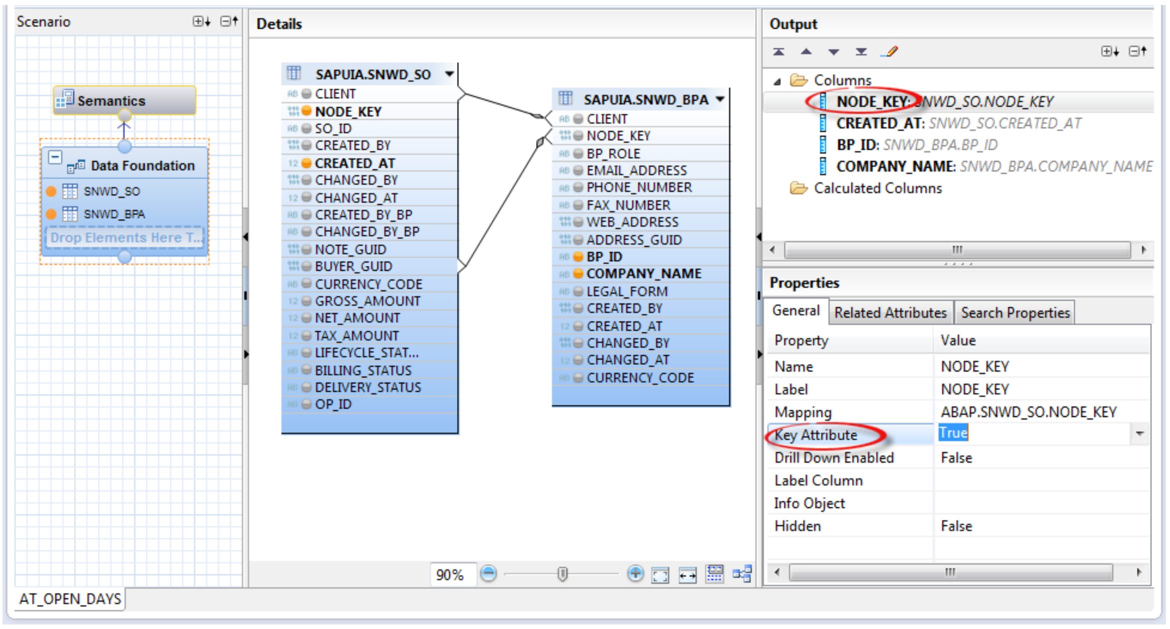
   Right-click the *Details* panel and choose context menu entry *Add to Output*.

   The selected table columns are now listed under *Columns* folder of the *Output* pane



NODE_KEY: SNWD_SO.NODE_KEY
CREATED_AT: SNWD_SO.CREATED_AT
BP_ID: SNWD_BPA.BP_ID
COMPANY_NAME: SNWD_BPA.COMPANY_

10. *Define SNWD_SO.NODE_KEY as key attribute.*
    Go to the *Output* pane of the *Data Foundation* node, select *column NODE_KEY* and set the property *Key Attribute* to *True* in the *General* tab of its *Properties* view.



NODE_KEY: SNWD_SO.NODE_KEY
CREATED_AT: SNWD_SO.CREATED_AT
BP_ID: SNWD_BPA.BP_ID
COMPANY_NAME: SNWD_BPA.COMPANY_NAME

Property	Value
Name	NODE_KEY
Label	NODE_KEY
Mapping	ABAP.SNWD_SO.NODE_KEY
Key Attribute	True
Drill Down Enabled	False
Label Column	
Info Object	
Hidden	False

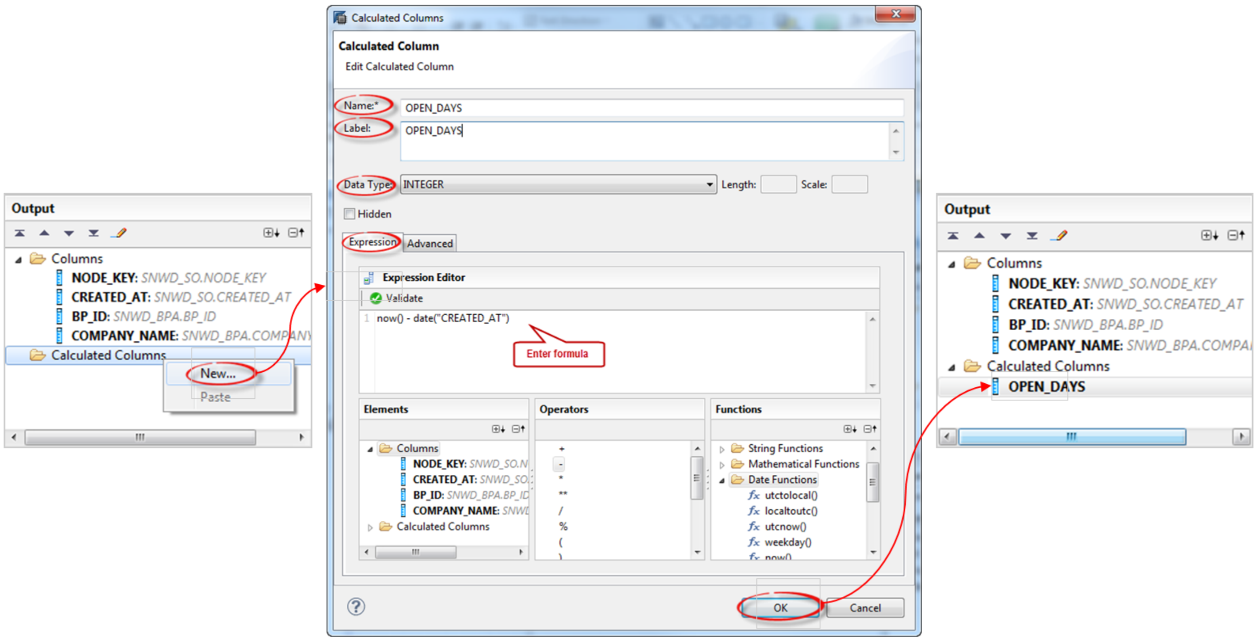
Note: A valid *Attribute View* requires at least one key attribute (column) to be defined. The key attribute is synonymous to the primary key of a table. It's important to define the key correctly since it identifies a unique row in an *Attribute View*.

11. Define a *Calculated* column named *OPEN_DAYS* which calculates the number of open days for a sales order.
    Go to the *Output* pane of the *Data Foundation* node, right-click the folder *Calculated Columns* and choose context menu entry *New...*.

    Enter *OPEN_DAYS* as name and maintain a description.

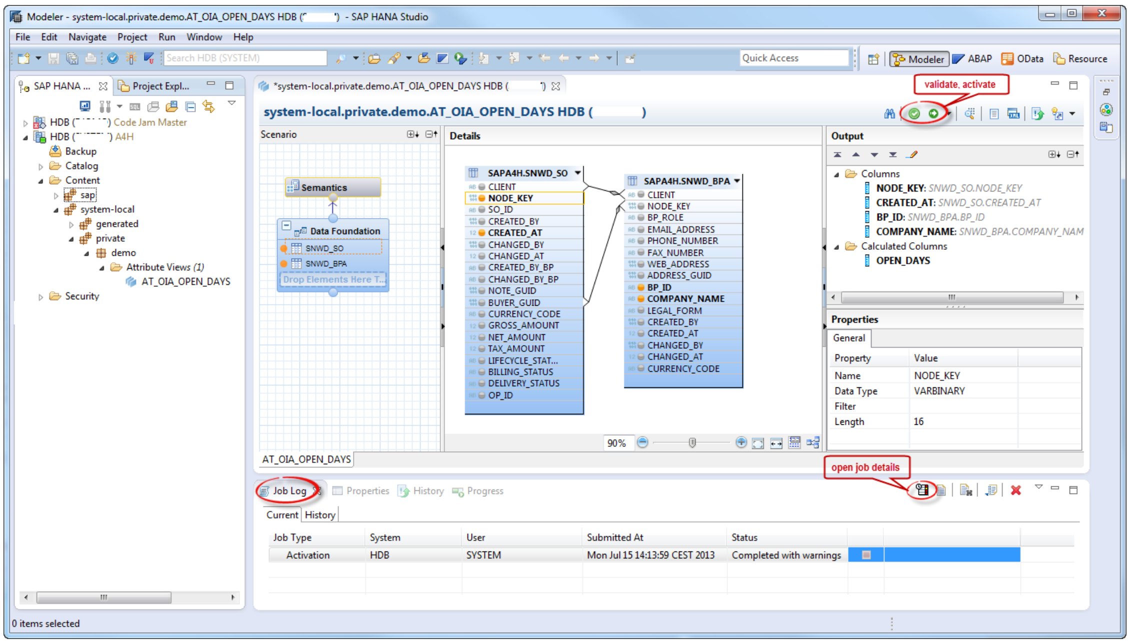
Choose *INTEGER* as data type and fill-in the formula for calculating the number of open days. To do so, you can either type-in the formula or drag and drop the relevant *Elements*, *Operators* .and *Functions* to the *Expression Editor*.
The formula is as follows: `now() - date("CREATED_AT")`

Press *OK*.



Note: The number of open days is determined by subtracting the date of order creation from current date. Function *now()* determines the current date and belongs to date functions. Function *date()* is a conversion function used to convert the creation timestamp of a sales order into date.

12. Save and activate your *attribute view* using the appropriate toolbar icon . Check if the activation was successful on the *Job Log* view.


Modeler - system-local.private.demo.AT_OIA_OPEN_DAYS HDB - SAP HANA Studio
File Edit Navigate Project Run Window Help
validate, activate
open job details
system-local.private.demo.AT_OIA_OPEN_DAYS HDB
Scenario
Details
Output
Semantics
Data Foundation
SNWD_SO
SNWD_BPA
AT_OIA_OPEN_DAYS
Job Log
Properties
History
Progress
Current
History

Job Type	System	User	Submitted At	Status
Activation	HDB	SYSTEM	Mon Jul 15 14:13:59 CEST 2013	Completed with warnings

You can display the log details by either double-clicking on the relevant entry or by selecting the entry and clicking on the *Open Job Details* toolbar icon

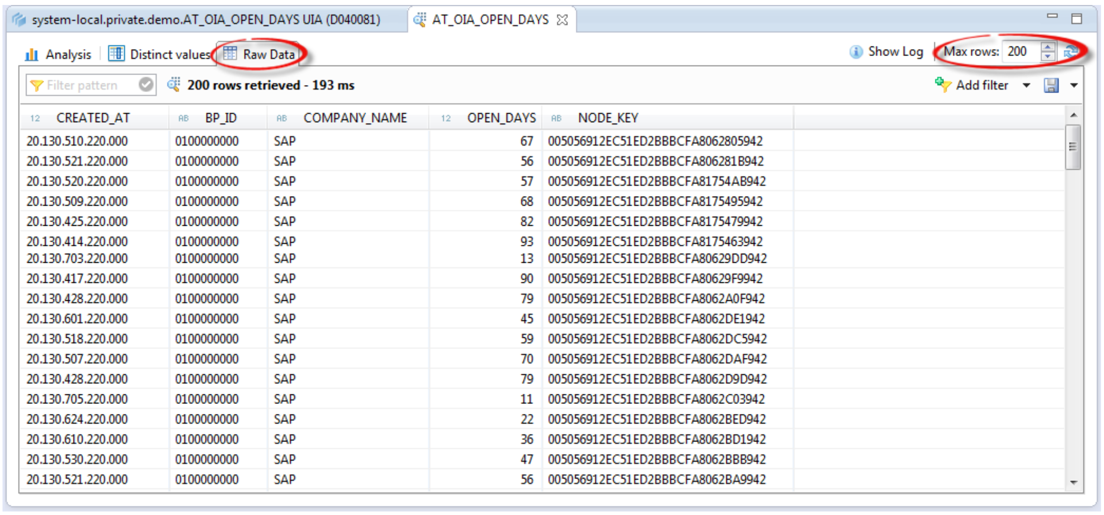
Client-side validation warnings under section *Large object data type check rule* regarding *VARBINARY* data type can be ignored for this exercise.

1. Preview the data retrieved using the *Attribute View* you just created.

   Press the *Data Preview* toolbar icon and choose the *Raw Data* tab on the next screen.

Info about the different *Data Preview* tab pages:

- *Raw Data*: All attributes along with data are displayed in a table format.
- *Distinct values:* All attributes along with data in a graphical format.
- *Analysis:* All attributes and measures in a graphical format.



200 rows retrieved - 193 ms

CREATED_AT	BP_ID	COMPANY_NAME	OPEN_DAYS	NODE_KEY
20.130.510.220.000	0100000000	SAP	67	005056912EC51ED2BBBCFA8062805942
20.130.521.220.000	0100000000	SAP	56	005056912EC51ED2BBBCFA806281B942
20.130.520.220.000	0100000000	SAP	57	005056912EC51ED2BBBCFA81754AB942
20.130.509.220.000	0100000000	SAP	68	005056912EC51ED2BBBCFA8175495942
20.130.425.220.000	0100000000	SAP	82	005056912EC51ED2BBBCFA8175479942
20.130.414.220.000	0100000000	SAP	93	005056912EC51ED2BBBCFA8175463942
20.130.703.220.000	0100000000	SAP	13	005056912EC51ED2BBBCFA80629DD942
20.130.417.220.000	0100000000	SAP	90	005056912EC51ED2BBBCFA80629F9942
20.130.428.220.000	0100000000	SAP	79	005056912EC51ED2BBBCFA8062A0F942
20.130.601.220.000	0100000000	SAP	45	005056912EC51ED2BBBCFA8062DE1942
20.130.518.220.000	0100000000	SAP	59	005056912EC51ED2BBBCFA8062DC5942
20.130.507.220.000	0100000000	SAP	70	005056912EC51ED2BBBCFA8062DAF942
20.130.428.220.000	0100000000	SAP	79	005056912EC51ED2BBBCFA8062D9D942
20.130.705.220.000	0100000000	SAP	11	005056912EC51ED2BBBCFA8062C03942
20.130.624.220.000	0100000000	SAP	22	005056912EC51ED2BBBCFA8062BED942
20.130.610.220.000	0100000000	SAP	36	005056912EC51ED2BBBCFA8062BD1942
20.130.530.220.000	0100000000	SAP	47	005056912EC51ED2BBBCFA8062BBB942
20.130.521.220.000	0100000000	SAP	56	005056912EC51ED2BBBCFA8062BA9942

The maximal number of rows to be displayed can be changed in field *Max rows*. Click the refresh button to update the displayed content.

Step 2: Expose a HANA View as External View in the ABAP Dictionary

We will now expose the previously created HANA view in the application server using the new ABAP dictionary feature called *External View*.

Note: The new ABAP dictionary features are only available in the *ABAP Development Tools for SAP NetWeaver* (aka *ABAP in Eclipse*) and not in the standard ABAP *Workbench* (transaction *SE80*).

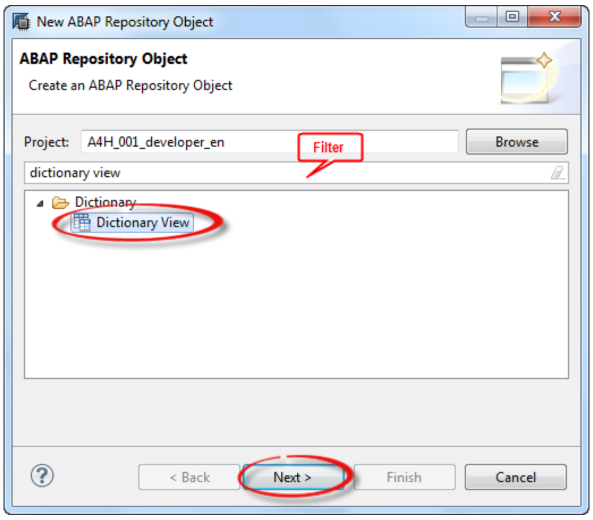
The procedure described below – *Exposing a HANA view as External View in the ABAP Dictionary*- is valid for all three types of SAP HANA views, meaning *Attribute Views*, *Calculation Views* and *Analytical Views*.

2. Open the ABAP perspective.

   Select menu entry *Window* > *Open perspective* > *Others…*, then choose ABAP from the dialog and press *OK*.

3. Create the dictionary view (View Type: *External View*) *ZAT_OIA_OPENDAYS* in the ABAP dictionary. Right-click the package of your choice and choose context menu entry *New* > *Other ABAP Repository Object. (*I will use my *$TMP* package for this purpose.)

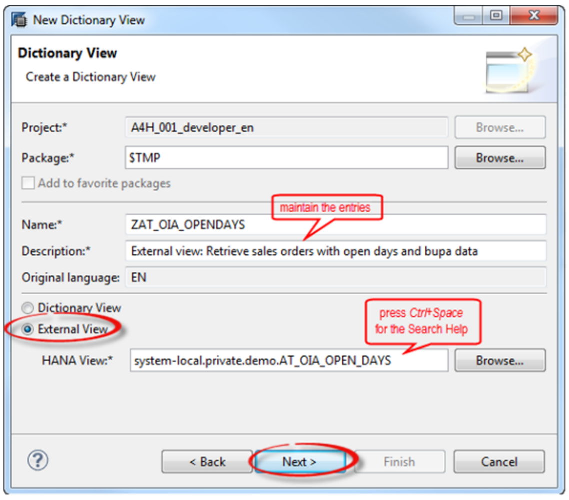
   Filter for "*Dictionary View*", choose the homonymous entry and press *Next*.



4. Enter a name (e.g. *ZAT_OIA_OPENDAYS*) and a description (e.g. "*External view: Retrieve sales orders with open days and bupa data*"). Choose the *External View* radio button, enter the name of the HANA *Attribute View* previously created in *Step 1* (e.g. *system-local.private.demo.AT_OIA_OPEN_DAYS*) and press *Next.*

   Tipp: Use shortcut *Ctrl+Space* for the content assistance (aka *Search Help*).

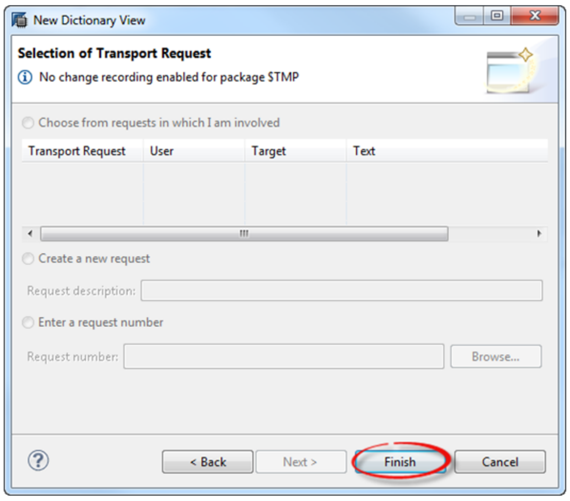
On the next dialog box, select a transport request if required and press *Finish*.



New Dictionary View

Dictionary View
Create a Dictionary View

Project:*	A4H_001_developer_en
Package:*	$TMP
Add to favorite packages	
Name:*	ZAT_OIA_OPENDAYS
Description:*	External view: Retrieve sales orders with open days and bupa data
Original language:	EN
Dictionary View	
[✓ External View]	
HANA View:*	system-local.private.demo.AT_OIA_OPEN_DAYS



New Dictionary View

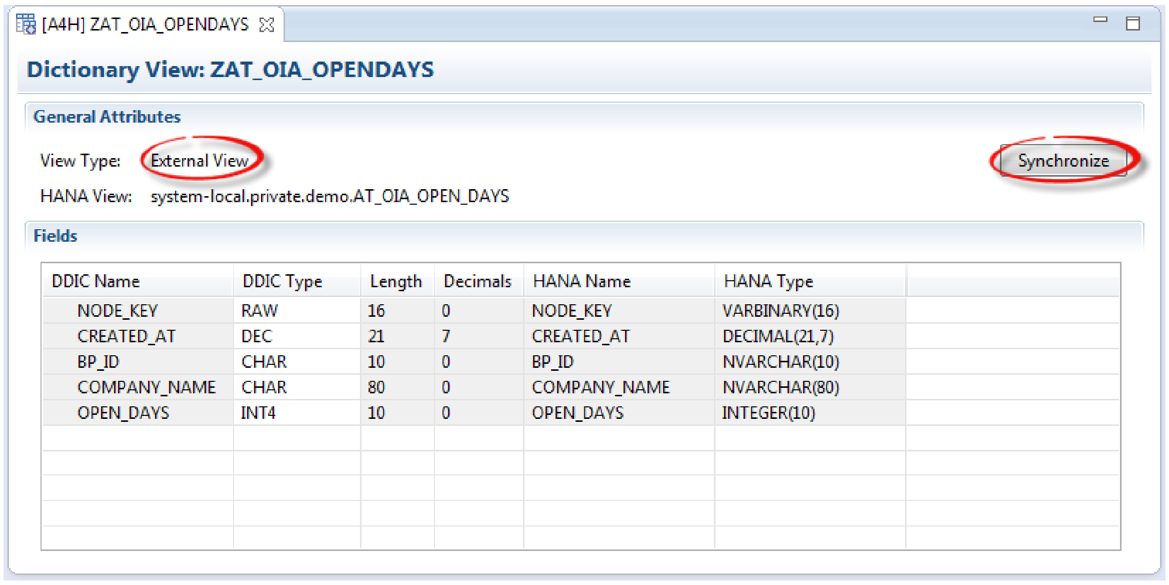
Selection of Transport Request
No change recording enabled for package $TMP

Choose from requests in which I am involved

Transport Request	User	Target	Text

Create a new request
Request description:
Enter a request number
Request number:

5. Save and activate the new *External View*.



[A4H] ZAT_OIA_OPENDAYS

Dictionary View: ZAT_OIA_OPENDAYS

General Attributes

View Type: External View
HANA View: system-local.private.demo.AT_OIA_OPEN_DAYS

Fields

DDIC Name	DDIC Type	Length	Decimals	HANA Name	HANA Type
NODE_KEY	RAW	16	0	NODE_KEY	VARBINARY(16)
CREATED_AT	DEC	21	7	CREATED_AT	DECIMAL(21,7)
BP_ID	CHAR	10	0	BP_ID	NVARCHAR(10)
COMPANY_NAME	CHAR	80	0	COMPANY_NAME	NVARCHAR(80)
OPEN_DAYS	INT4	10	0	OPEN_DAYS	INTEGER(10)

Information about *External View*
An External View only represents the HANA view from which it is derived. In particular, this means that the HANA view is the leading object: changes made to the fields in the view in the HANA Repository imply changes for the external view (in ABAP Dictionary) and all consumers.

Information about synchronizing contents with the SAP HANA Repository
in the *External View*, press the *"Synchronize*" button to reload the metadata of the underlying HANA view if it was changed in the HANA repository. Do not forget to save and activate the updated dictionary object afterwards.

6. You can preview an *external view* in the ABAP dictionary (*SE11* and *SE16*) like you do for any other classical dictionary view or database.

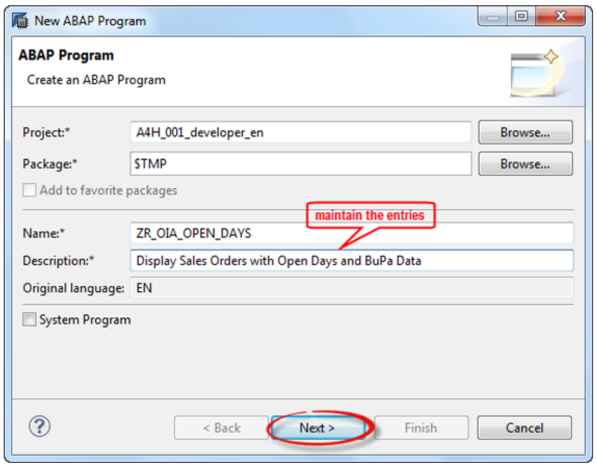
Step 3: Implement an ABAP Report consuming an External View

We will now create and implement a simple ABAP report which read the resulting dataset via the *External View* in *openSQL* and display the resulting dataset using the standard ALV.

1. Create ABAP program *ZR_OIA_OPEN_DAYS.*
   Select the package of your choice, right-click on it and choose context menu entry *New > ABAP Program*.

2. Enter a name (e.g. *ZR_OIA_OPEN_DAYS*), a description (e.g. "Retrieve and *Display Sales Order with Open Days and BuPa Data*") and press *Next.*

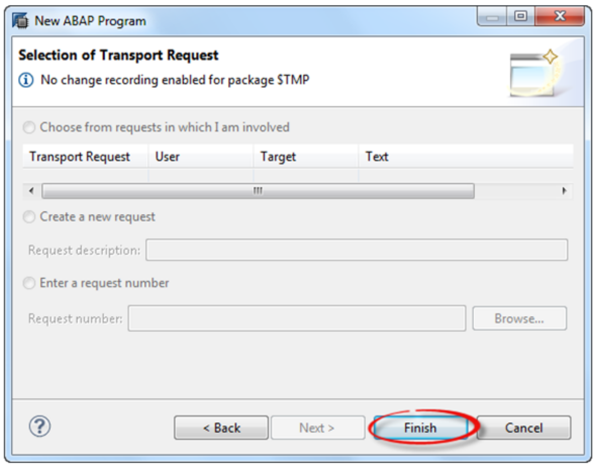
   On the next dialog screen, select a transport request if required and press *Finish.*



New ABAP Program

ABAP Program
Create an ABAP Program

Project:* A4H_001_developer_en
Package:* $TMP
Add to favorite packages
Name:* ZR_OIA_OPEN_DAYS
Description:* Display Sales Orders with Open Days and BuPa Data
Original language: EN
System Program



New ABAP Program

Selection of Transport Request
No change recording enabled for package $TMP

Choose from requests in which I am involved

Transport Request	User	Target	Text

Create a new request
Request description:
Enter a request number
Request number:

3. Implement the report.
   Copy & paste the source code below into the ABAP editor.

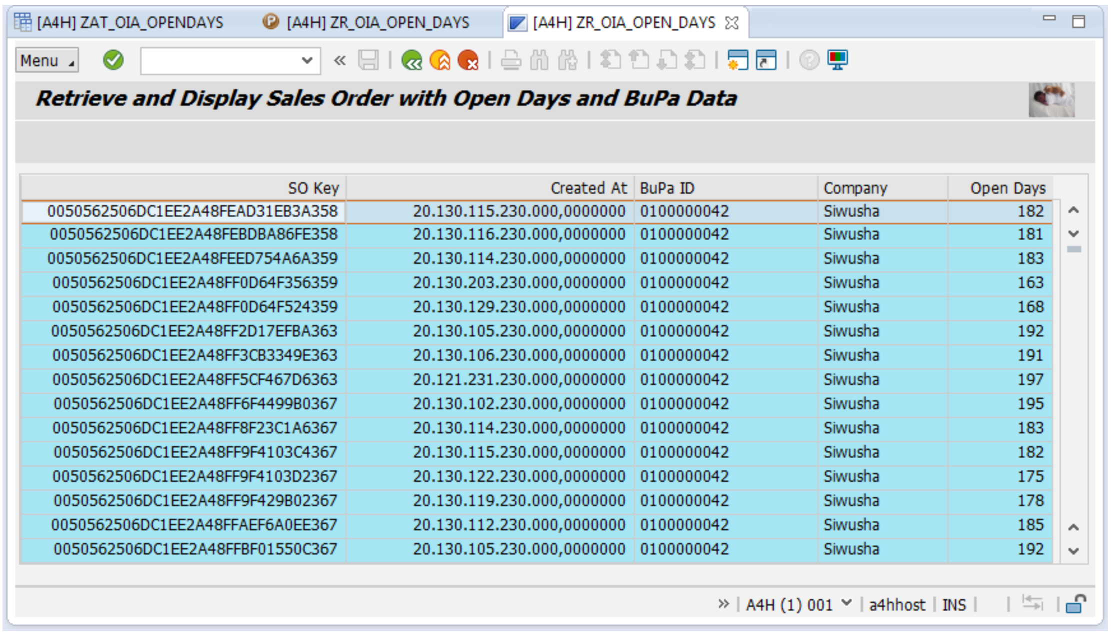
Report *ZR_OIA_OPEN_DAYS*

```
REPORT zr_oia_open_days.

DATA lt_open_days   TYPE STANDARD TABLE OF zat_oia_opendays.

* Read the sales order via the external view using Open SQL
SELECT * FROM zat_oia_opendays INTO TABLE lt_open_days.

* display the resulting dataset of sales orders
* (All columns are not displayed for simplification reasons)
WRITE: / 'Sales Order Key                 ', 'Customer ID     ', 'Open Days'.
LOOP AT lt_open_days ASSIGNING FIELD-SYMBOL(<f>).
  WRITE:/ <f>-node_key, <f>-bp_id, <f>-open_days .
ENDLOOP.
```

Note: Analogous to other regular dictionary views, external views are called using the *SELECT* statement (*Open SQL*). Meaning no special ABAP statement has to be called like it is the case for *Database Procedure Proxy* entities where statement *CALL DATABASE PROCEDURE* has been added to the ABAP language.

4. Save and activate your report

5. You can now run the report (press *F8*) and see the result of your effort.



Retrieve and Display Sales Order with Open Days and BuPa Data

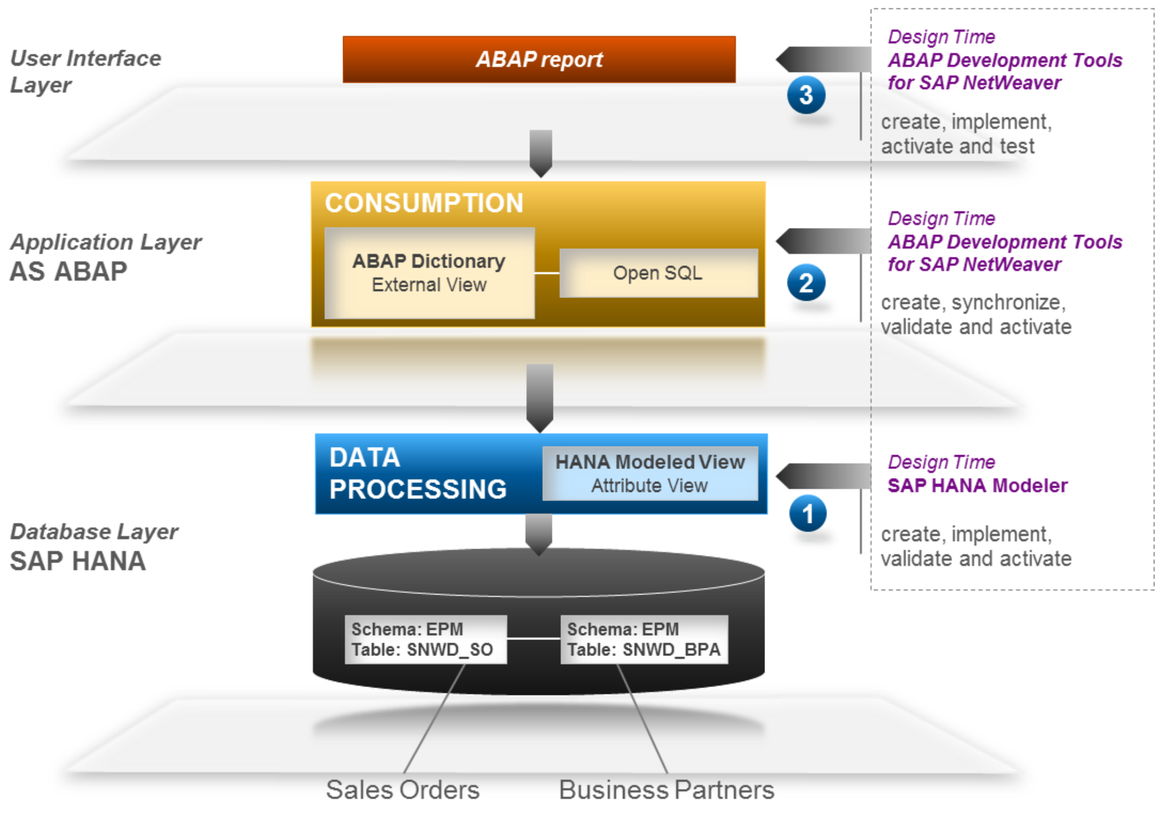
SO Key	Created At	BuPa ID	Company	Open Days
0050562506DC1EE2A48FEAD31EB3A358	20.130.115.230.000,0000000	0100000042	Siwusha	182
0050562506DC1EE2A48FEBDBA86FE358	20.130.116.230.000,0000000	0100000042	Siwusha	181
0050562506DC1EE2A48FEED754A6A359	20.130.114.230.000,0000000	0100000042	Siwusha	183
0050562506DC1EE2A48FF0D64F356359	20.130.203.230.000,0000000	0100000042	Siwusha	163
0050562506DC1EE2A48FF0D64F524359	20.130.129.230.000,0000000	0100000042	Siwusha	168
0050562506DC1EE2A48FF2D17EFBA363	20.130.105.230.000,0000000	0100000042	Siwusha	192
0050562506DC1EE2A48FF3CB3349E363	20.130.106.230.000,0000000	0100000042	Siwusha	191
0050562506DC1EE2A48FF5CF467D6363	20.121.231.230.000,0000000	0100000042	Siwusha	197
0050562506DC1EE2A48FF6F4499B0367	20.130.102.230.000,0000000	0100000042	Siwusha	195
0050562506DC1EE2A48FF8F23C1A6367	20.130.114.230.000,0000000	0100000042	Siwusha	183
0050562506DC1EE2A48FF9F4103C4367	20.130.115.230.000,0000000	0100000042	Siwusha	182
0050562506DC1EE2A48FF9F4103D2367	20.130.122.230.000,0000000	0100000042	Siwusha	175
0050562506DC1EE2A48FF9F429B02367	20.130.119.230.000,0000000	0100000042	Siwusha	178
0050562506DC1EE2A48FFAEF6A0EE367	20.130.112.230.000,0000000	0100000042	Siwusha	185
0050562506DC1EE2A48FFBF01550C367	20.130.105.230.000,0000000	0100000042	Siwusha	192

Summary

Congratulations! You have just experienced how easy it is to consume a HANA view – *Attribute View*, *Calculation View or Analytical View* - from an ABAP programs using the new DDIC entity called *External View*.

You can now create a HANA (Attribute) View, expose it as *External View* in the ABAP dictionary and access the view natively in your ABAP program.

The illustration below provides the high level architecture underlying this tutorial.


User Interface Layer
ABAP report
Design Time
ABAP Development Tools for SAP NetWeaver
3
create, implement, activate and test
Application Layer
AS ABAP
CONSUMPTION
ABAP Dictionary External View
Open SQL
Design Time
ABAP Development Tools for SAP NetWeaver
2
create, synchronize, validate and activate
DATA PROCESSING
HANA Modeled View Attribute View
Design Time
SAP HANA Modeler
1
create, implement, validate and activate
Database Layer
SAP HANA
Schema: EPM Table: SNWD_SO
Schema: EPM Table: SNWD_BPA
Sales Orders
Business Partners

Related Content

ABAP for SAP HANA Reference Scenario, SCN document, http://scn.sap.com/docs/DOC-35518

SAP HANA Developer Guide, SAP Help Guide, http://help.sap.com/hana/hana_dev_en.pdf

SAP HANA SQLScript Reference, SAP Help Guide, http://help.sap.com/hana/hana_dev_sqlscript_en.pdf

Find more content under SCN space *ABAP for SAP HANA*, http://scn.sap.com/community/abap/hana

www.sap.com