**Open ODS View**

[Содержание](#Содержание)

# [How to go Virtual in BW Part I - The Open ODS View](#The_Open_ODS_View)

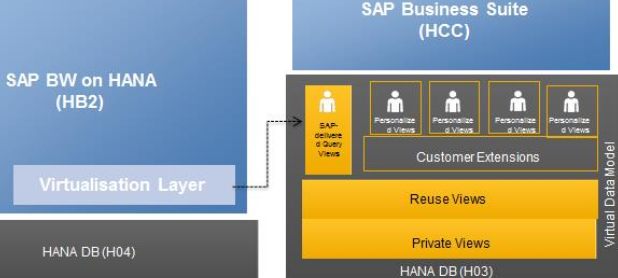
<https://blogs.sap.com/2014/10/17/open-ods-view-in-hana-edw-the-best-of-bw-and-hana-combined/>

Since the introduction of SAP HANA many customers wish to add more robust features that they are familiar with from SAP BW into HANA. For those customers now they can adopt BW on HANA - a SQL-oriented data warehouse - or a combination of both which essentially consists of *SAP BW and SQL-minded approach termed HANA* ***EDW***. Especially if you have implemented ERP or CRM etc. on HANA - you can start from SAP HANA with a SQL approach for Operational Reporting and move into the SAP BW side in a virtual fashion for data warehousing.

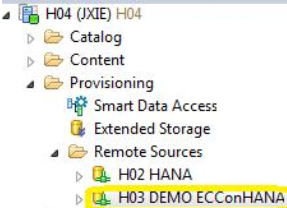
With BW 7.4 SP5+ we have a new *acquisition* (ˌækwɪˈzɪʃən приобретение | владение) layer - *Open ODS based on fields rather than InfoObjects* to allow for *Function before Integration’modelling*. Prior to BW 7.4 nearly all of the common function modelling was based on InfoObjects | BEx Queries | InfoProviders | Dataflows | Authorisations and Operations. It’s the InfoObjects that define the integrated EDW model and hence *no InfoObjects no functions*. On the contrary with ***Function before Integration modelling*** the Open ODS field-based approach makes the following possible -

* Function modelling on any field data
* Crossing the border - *Open ODS Function modelling on remote data* - ***HANA Smart Data Access***.
* Integration of field data into EDW context – association of InfoObjects to Open ODS Views
* BEx Queries atop Open ODS Views
* Embed Open ODS Views into Composite Providers - only union is allowed for Open ODS views up to SP7. Join is supported as of SP8.

What does this mean in practice? We will be demonstrating the technical capability of Open ODS Function modelling on remote data. In the NTT DATA landscape - we have BW on HANA DB - namely H04 accessing a remote ECC on HANA DB - namely H03 which has the HANA Live Virtual Data Models installed. The landscape is illustrated below

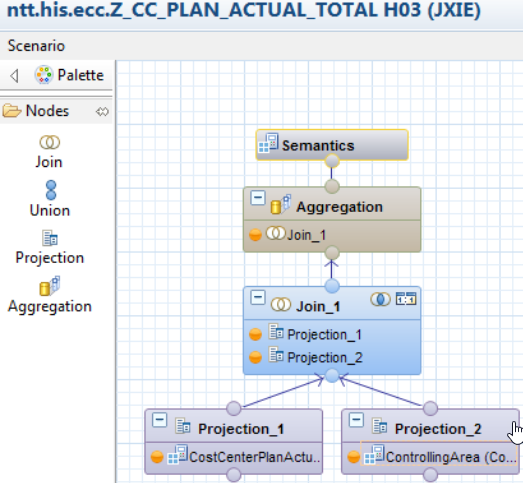


This landscape is represented in the HANA Studio as follows

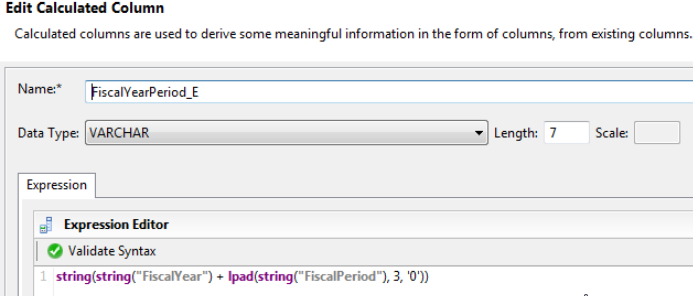


The scenario we are demonstrating is Cost Centre Plan and Actual Totals.

In H03, we have created our own custom Calculation View Z\_CC\_PLAN\_ACTUAL\_TOTAL with a simple join of 2 HANA Live Reuse Views - CostCenterPlanActualCostTotals and ControllingArea



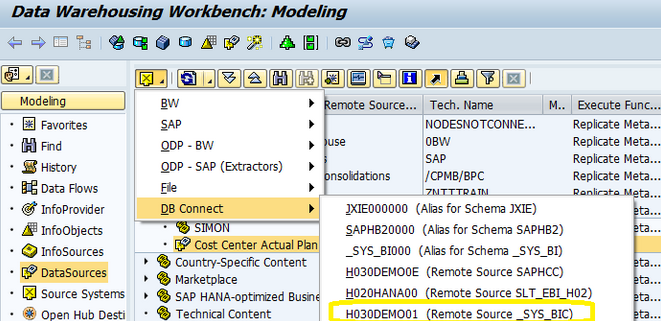
In the node Projection\_1, we created a calculated column for FiscalYearPeriod



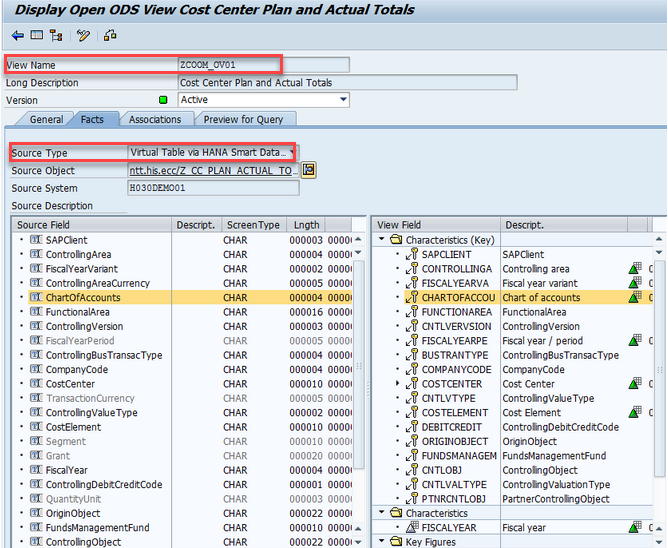
**LPAD** (*<str>*, *<n>* [, *<pattern>*]) - left-pads a string with spaces, or a specified pattern, to make a string of a specified number of characters in length.

In the next step let’s move to BW (HB2) to create an Open ODS View which will allow us to consume/expose the HANA Calculation View Z\_CC\_PLAN\_ACTUAL\_TOTAL to BW (Please note that it’s also possible to do that within BW Modelling Tools, which is a plugin installed on Eclipse).

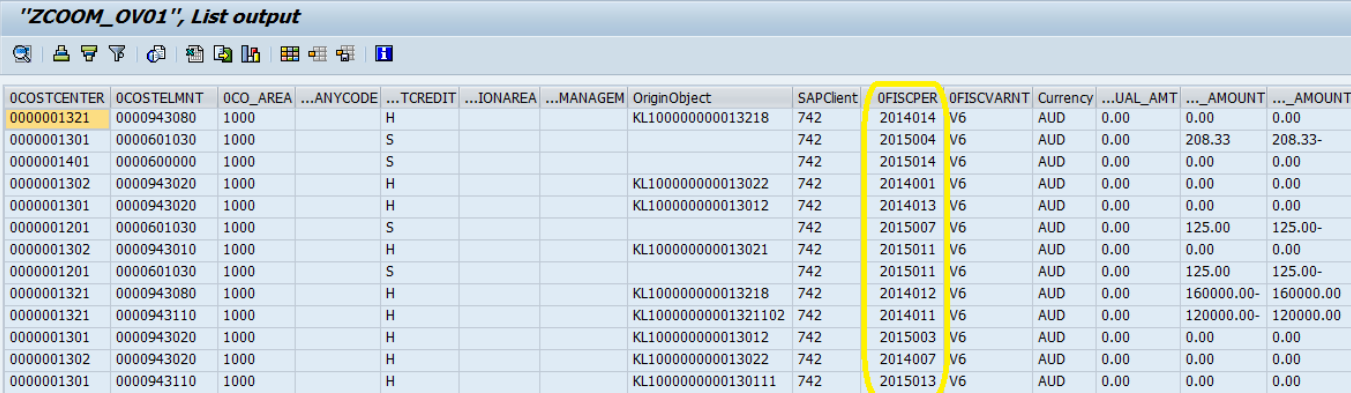
*Prior to the creation of an Open ODS View, make sure a DB Connection is available*



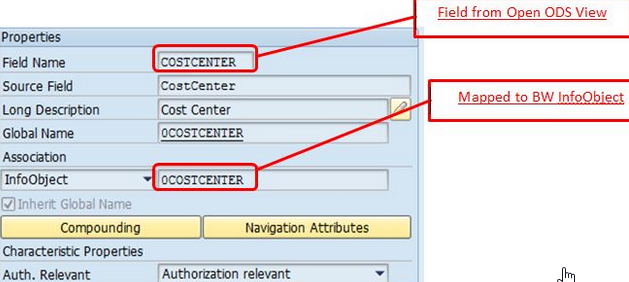
Then we have created an Open ODS View ZCOOM\_OV01 with the type **Virtual Table via** ('vaɪə) **HANA Smart Data Access** based on our Calculation View



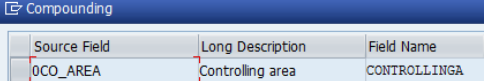
By using the ***Display Data*** functionality, we see that the FiscalYearPeriod field is correctly populated via the calculated column in my CalculationView. In fact, all the data available in the remote HANA system H03, is now available as an InfoProvider in BW



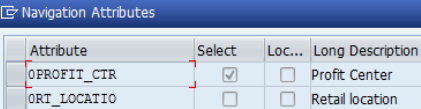
*To integrate fields into an EDW context, we have associated fields with InfoObjects.* It is also possible to associate fields with Open ODS Views and the purpose of InfoObjects association is to leverage some of the features offered by BW Analytic Manager (aka OLAP Engine), for example hierarchy support and InfoObjects’ Navigational Attributes. Below is an illustration of association to 0COSTCENTER



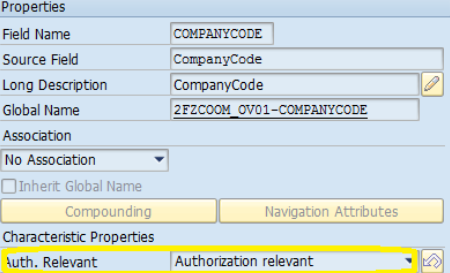
As 0COSTCENTER is compounded to 0CO\_AREA, we also specify the compounding (объединение, 'kɔmpaund составление) for COSTCENTER

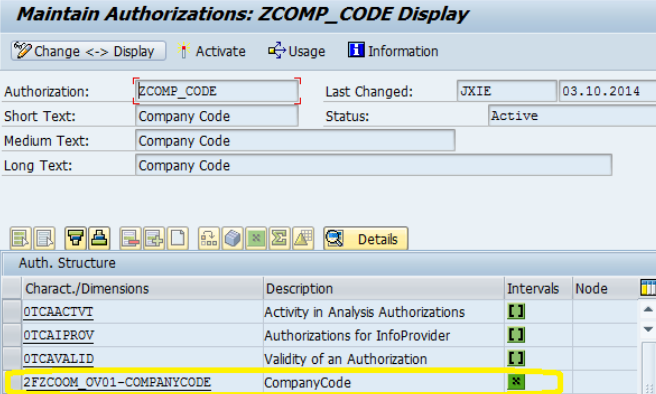


In addition, 0PROFIT\_CTR is added as a navigational attribute

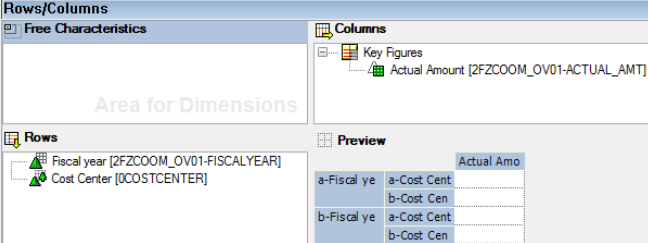


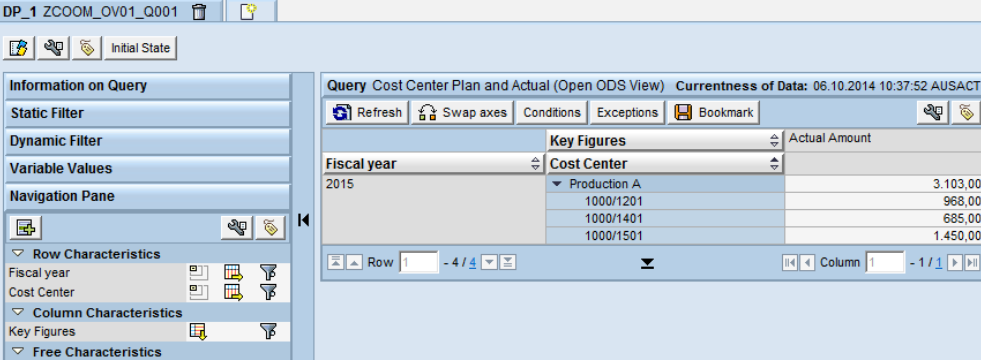
If a field is assigned to an InfoObject, BW’s Analysis Authorisation (полномочия на анализ) can be leveraged ('liːv(ə)rɪʤ использовать) as (могут использоваться в качестве) ‘Authorisation relevancy’ of the InfoObject is inherited. Whereas if a field is not assigned to any InfoObject, BW also supports creating Analysis Authorisation directly based on the field. *Below is an Analysis Authorisation created on the field CompanyCode*





In the final step, a BEx Query is created on top of the Open ODS View with the standard Cost Centre hierarchy enabled for display, which saves you from dealing with hierarchies within HANA. Please note that fields are also supported on BEx Query





Whereto from here?

Starting from SAP BW 7.4 you could embed Open ODS Views in a CompositeProvider (Latest version only allows this functionality via BW modelling perspective in Eclipse). Unfortunately this allows you to do only Union if you are not on SP8. *Another limitation is that you cannot create an Open ODS View on most HANA Live Query Views due to their input parameters.* With BW 7.4 SP8 becoming generally available it provides the following exciting features

* Support of planning functionalities for UNION cases
* Possibility to create a CompositeProvider on top of Composite Providers (only UNION supported)
* Use of Open ODS views as Source Objects and allowing JOINS to be used
* Support of input parameters when Open ODS views are used as source objects
* We will be providing more demos based on the new functionalities introduced in SP8.

# How to go Virtual in BW Part I - The Open ODS View

<https://www.just-bi.nl/virtual-bw-open-ods-view/>

[Содержание](#Содержание)

### **The Age of Virtual Reality**

In the days of old when Wi-Fi wasn’t readily available and Google was nothing but a misspelling of the word googly, *BW could not join virtual InfoProviders based on external data*. Not any longer. Why? Because of the Open ODS View (OODS) & the HANA CompositeProvider (HCPR). How much storage space does it take to store both these InfoProviders? The answer - virtually nothing (pun (pʌn игра слов, каламбур, уплотнять бетон трамбовкой) intended).

These virtual InfoProviders are not new and neither is their platform, SAP HANA. The OODS & HCPR have been available since BW 7.40. To use them you need to [install Eclipse and the BW Modeling Tools](https://www.just-bi.nl/installing-necessary-tools-use-bw4hana/). After that, you can start modeling them in the BW Modeling Perspective (instead of RSA1). However, as an SAP BI consultant I’ve noticed that most customers have often either

1) Made the step to HANA, without utilizing the new modeling possibilities or

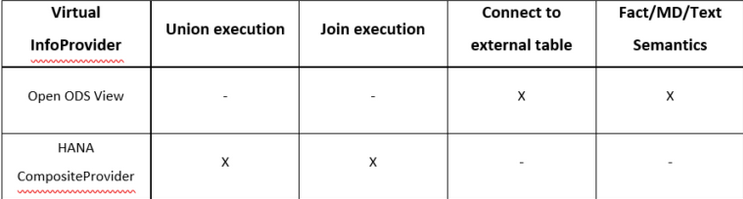
2) Not yet made the step to HANA at all.

Furthermore, amongst my fellow consultants I have noticed a strong movement towards discovering the full potential of virtual modeling and HANA. Part I and [Part II of this article](https://www.just-bi.nl/virtual-bw-hana-compositeprovider/) will give you a general idea about how the Open ODS View and CompositeProvider work, so you can understand their potential.

### **What are the OODS & HCPR and what are they not?**

When I talk to people about the Open ODS View and the CompositeProvider, the first thing they usually ask is - what’s the difference?

The difference between the two InfoProviders can be summed up as follows



*Table 1 - The difference between an Open ODS View and a CompositeProvider*

From Table 1 we can gather that although they are both virtual objects, we use them for different purposes. Usually when I hear somebody describing a CompositeProvider they might say - A CompositeProvider is like a MultiProvider. However, this is only partially true. The CompositeProvider and the MultiProvider we know from older BW versions can both execute Unions. The CompositeProvider is more advanced though, as we can gather from Table 1, because it can also perform Joins (like the old InfoSet). The difference between a Join and Union? Simply said, a Union adds the rows of two datasets on top of eachother. A Join adds columns of one table to another based on matching values in shared columns.

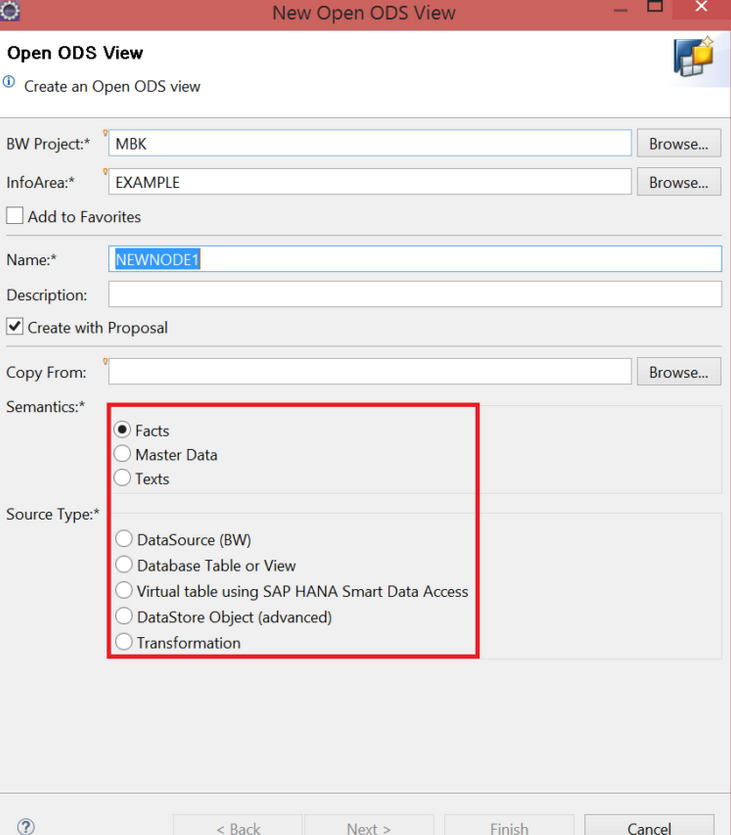
If the CompositeProvider is so advanced, why then use an Open ODS View? The main reason is the *Open ODS View’s ability to connect to external tables*. Let’s say you have data sitting on an external database that you want to use in BW, however you don’t want to persist it. *You can then set up a* ***SDA (Smart Data Access) Connection*** *to that database and create a* ***Virtual Table****.*

*After setting up an SDA connection and creating a Virtual Table, you can create an Open ODS View based on that connection* and voilà - you have a virtual connection to that table. Any query on this Open ODS View will then connect to an external table at runtime. However, the query will be dependent on the external system, the connection to the external system and the amount of data pulled in, for performance and availability. Querying external data at runtime can save space though (and therefore money). For those reasons, you need to think about both cost and performance, when choosing whether or not to use an OODS.

In the following sections, I’ll walk you through the Open ODS View creation process. In [Part II of this article](https://www.just-bi.nl/virtual-bw-hana-compositeprovider/) I’ll discuss the HANA CompositeProvider and its properties.

### **Open ODS View Source Types**

There are a lot of options to pick from when creating an OODS, as we can see from Figure 1 below. I’ll (briefly) discuss each of the 5 different Source Types and Semantics. I’ll start with the Source Types.

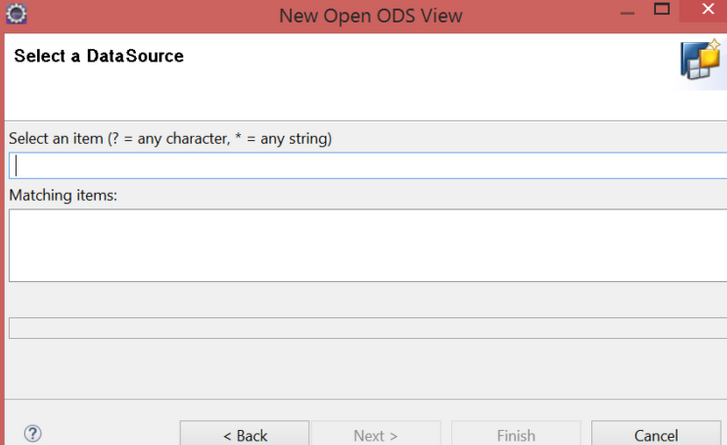


*Figure 1. The 1st screen you see when creating an Open ODS View*

#### **1. Source Type - DataSource (BW)**

Let’s start off with the first Source Type option. The DataSource (BW) option refers to the DataSources you probably know all too well, if you have prior experience with BW. Using these DataSources, you can connect to an underlying SAP ECC system for instance. *You need to have first created and configured the DataSources first*, of course. Also, the *OODS does not accept File-based DataSources*.

Creating an Open ODS View using this option is quite straightforward. First you right-click one of your InfoAreas in Eclipse, or right-click your [Graphical Data Flow Modeler](https://www.just-bi.nl/say-goodbye-to-old-school-bw-data-flows/) if you’re using [BW/4HANA,](https://www.just-bi.nl/bw4hana-fantastic-journey/) and choose to create an OODS. After selecting DataSource (BW) and a Semantics type (any type will do for now), you press “Next”.

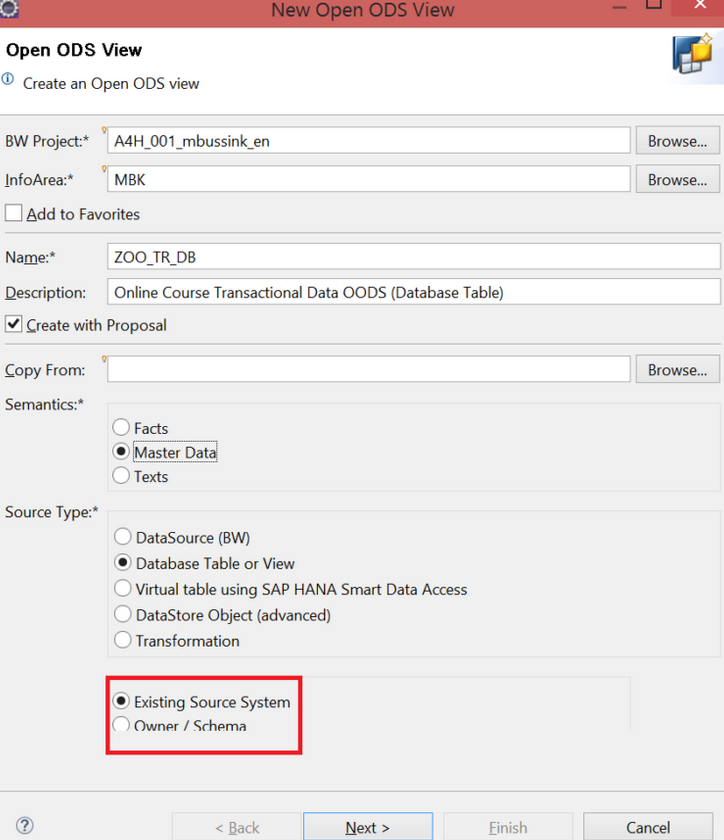


*Figure 2. Second screen when creating an Open ODS View*

After pressing “Next”, all you have to do is select the DataSource you want (see Figure 2) and you’ve finished setting up the Source Type!

#### **2. Source Type - Database Table or View**

Database Table or View is the option you choose *when you want to build an Open ODS View on a table residing in your local HANA database*. Right off the bat (bæt бита, летучая мышь) we can see there’s something distinctly different in Figure 3.

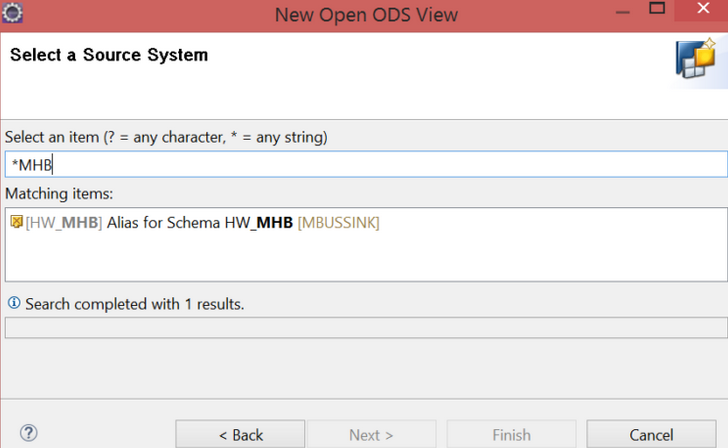


*Figure 3. Extra options appear when you select Database Table or View*

We are prompted by the screen to choose between Existing Source System and Owner / Schema. What could that possibly mean? I’ll explain

##### **Existing Source System**

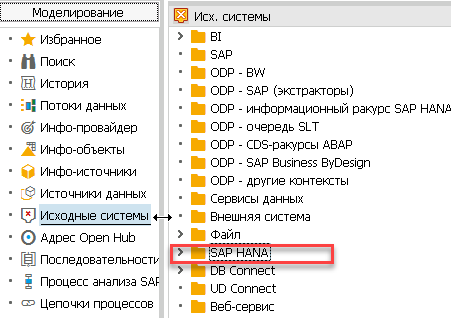
When you select Existing Source System the system checks whether you have a HANA Source System (see Figure 4). *A* ***HANA Source System*** *points towards a specific schema in your HANA Database.* HANA Source Systems let you use any tables in that schema you want for your Open ODS View.



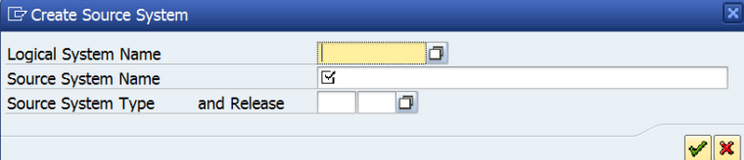
*Figure 4. Select a Source System screen*

##### **How to create a HANA Source System**

If your first reaction to the words “HANA Source System” is one of be’wilderment ('bɪ'wɪldəmənt замешательство) and confusion, no worries - I’ll explain how to create one. First, head to transaction RSA1 in the SAP GUI and go to Source Systems. Then, right-click your SAP HANA folder and select Create.

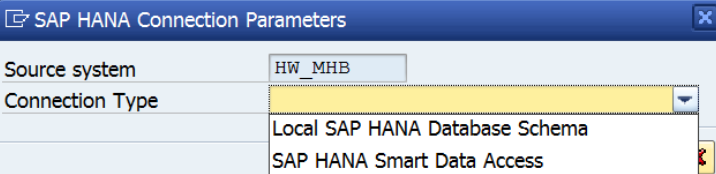


You’ll then see the following screen



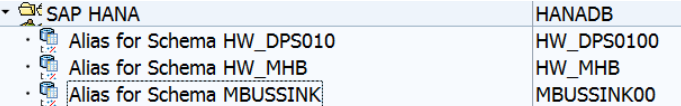
*Figure 5. Create Source System screen*

Enter a Logical System Name, which will be the technical name of your Source System. You could for instance give it the name of your schema, so you know with which schema it connects. Also, you need to give your new Source System a description. After that, press the green button.



*Figure 6. SAP HANA Connection Parameters*

Now we’re only one step away from completing our Source System. In Figure 6 we see the different possible Connection Types. If you have a local SAP HANA database schema, you choose the first option. If you want to set up a Smart Data Access connection to an external table, you choose the second. *Because* ***Database Table or View*** *concern local HANA tables, we’re going to pick* ***Local SAP HANA Database Schema****.*

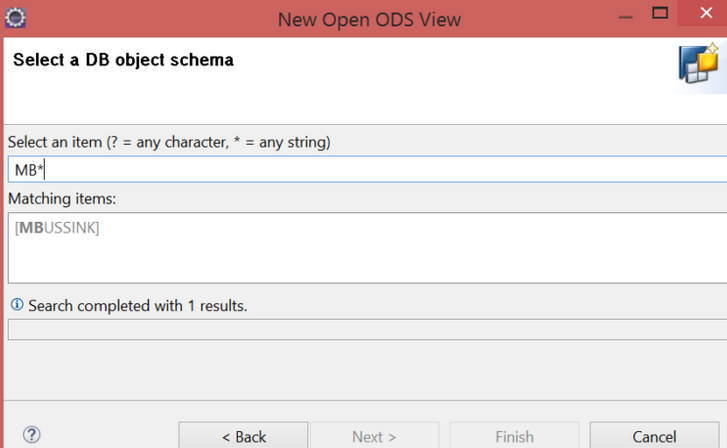


*Figure 7. After pressing the green arrow, your new Source System will appear in the SAP HANA folder*

Now you can pick any table in the schema your Source System points to, when creating an Open ODS View based on a database table or view. You simply select the Source System you just made and press Next. On the next screen you’ll be able to pick any table in the schema your Source System points to.

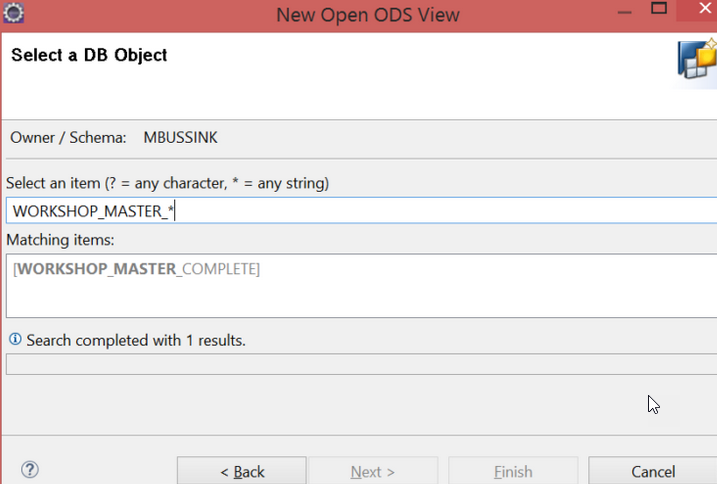
##### **Owner / Schema**

Whenever you select **Owner / Schema** on the first OODS creation screen, the next screen will ask you to select a schema.



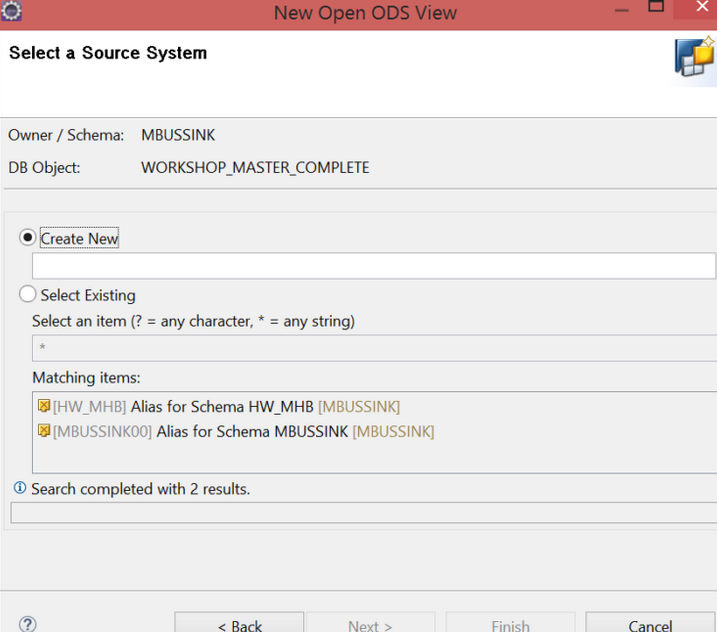
*Figure 8. Select a DB object schema*

After you’ve selected a schema, you need to choose the table you want



*Figure 9. Select a DB Object*

After we press “Next” in Figure 9, we can tell the difference between the “Existing Source System” and “Owner / Schema” options (see Figure 10)



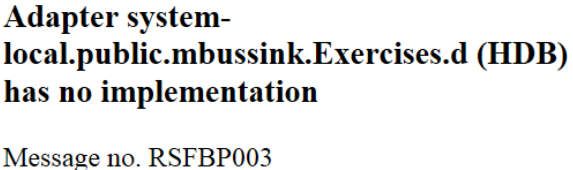
*Figure 10. Select a Source System*

You can either select an existing Source System or create a new one in Figure 10. Either way, *you’re going to need a HANA Source System, before you can start modeling your OODS*.

Conclusion - the “Existing Source System” and “Owner / Schema” options are basically 2 sides of the same coin.

1. When you pick “Existing Source System”, you pick a Source System that points to a particular schema and then you pick the table you want.
2. When you pick “Owner / Schema”, you first pick a schema and then pick the table you want. After that you can select an existing Source System that points to the schema you chose. If you haven’t created a Source System for that schema yet, you can create a Source System on the spot.

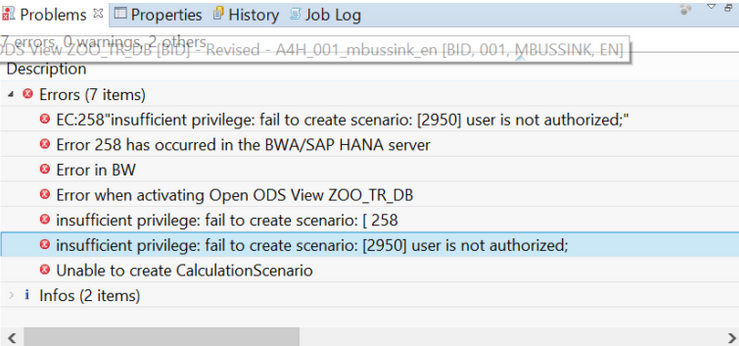
**Dis’claimer** (dɪs'kleɪmə оговорка) - the automatic Source System creation option does not always work, I got an error message (see Figure 11). If creating a Source System via the Owner/Schema option does not work, follow the instructions that accompany Figure 5 – 7 to create one manually.



*Figure 11. Error I got when trying to create a HANA Source System automatically via OODS*

##### **Privileges**

A*fter you create an Open ODS View based on a database table or view, you might see the following error*



*Figure 12. Insufficient privilege*

If you get this error *you need to grant your BW system user rights to whatever schema you want to build an OODS on*. You do so by opening the SQL Editor in Eclipse and using the following statement in Figure 13 (don’t forget to replace my schema with yours)



*Figure 13. How to grant Select privileges to the BW user in HANA*

If you wrote the statement correctly, you should see the message in Figure 14



*Figure 14. Select privileges successfully granted*

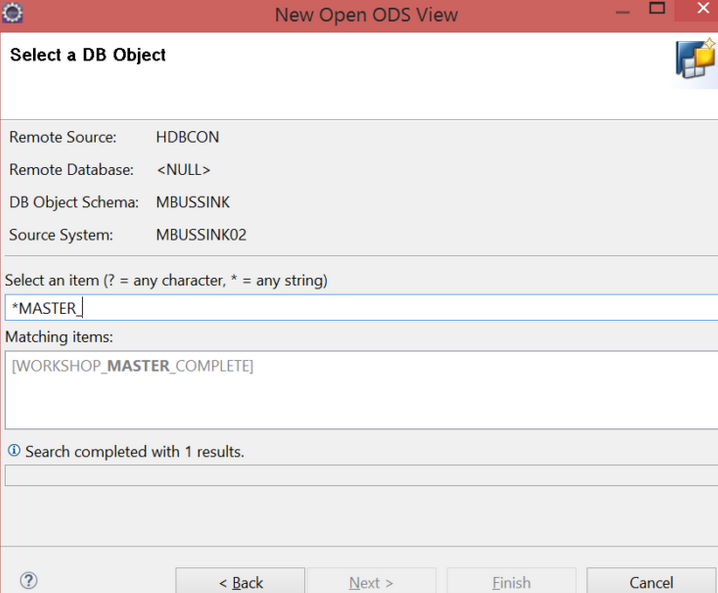
After that you should be able to start modelling and finish creating your OODS, without worrying about privileges!

#### **3. Source Type - Virtual table using SAP HANA Smart Data Access**

This is the option you choose *when you want to connect to an external table*, such as a Hadoop table, *without using a DataSource*. Creating an Open ODS View using a virtual table is very similar to creating one based on a database table or view.

##### **Existing Source System**

If you’ve already created a Source System that points to the right schema, you simply select “Existing Source System” and pick the table you want (see Figure 15).

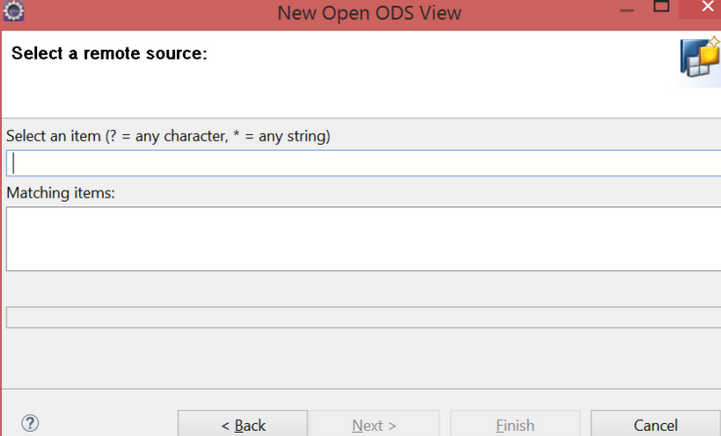


*Figure 15. Selecting a table from a Remote Source*

If there’s no Source System yet, you can create one as we did in Figure 5-6. Instead of selecting “Local SAP HANA Database Schema”, this time you select “SAP HANA Smart Data Access”. I here’by (ˌhɪə'baɪ этим, *таким образом*) assume that you’ve already set up a working connection to a Remote Source, which is outside of the scope of this article. Also, remember that you need to have the necessary privileges, otherwise you’ll get an error while creating a Remote Source. *You need the CREATE REMOTE SOURCE system privilege.* Also, *you need to create a Virtual Table on the table you want to base this OODS on*. For that, you need the object privilege *CREATE VIRTUAL TABLE*.

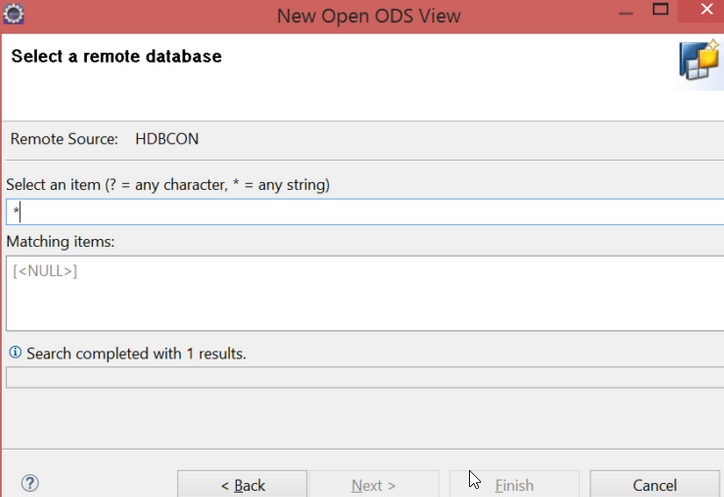
##### **Owner / Schema**

If you select the “Owner / Schema” option on the first screen, you’ll be prompted to select a Remote Source on the second (See Figure 16).



*Figure 16. Select a remote source*

This remote source can be Hadoop for instance. Once you click next you’ll have to select a remote database



*Figure 17. Select a Remote Database*

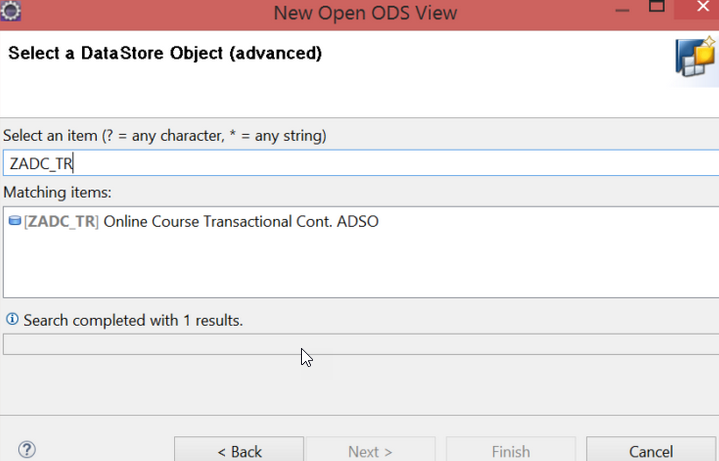
Depending on your Remote Connection, you might end up in a situation comparable to Figure 17 where your only option is [<NULL>]. *No reason for panic! If you see this just select [<NULL>], then press Next and you’ll be able to continue.* *In the next screen, you’ll have the option to create a new Source System or use an existing one* (comparable to Figure 11). After that you provide the semantics to your fields (which I will explain later).

#### **4. DataStore Object (advanced)**

You can base an Open ODS View on [an ADSO](https://www.just-bi.nl/all-you-need-to-know-about-adso/) using this option. Open ODS Views are generally used to pull in data from external systems. However, *the ADSO Source Type can still be of use if you want to specify whether a field-based ADSO contains text, master data or fact data*. Field-based means - fields modeled without InfoObjects.

You might wonder - why would you want to do that? By basing an OODS on an ADSO, you can define that ADSO as Master Data. Defining an ADSO as Master Data lets you use the underlying ADSO as Master Data for Fact-table type Open ODS Views and CompositeProviders. Also, if you want to convert the data type of your ADSO fields, you can use an OODS View on an ADSO to convert the fields by associating them with an InfoObject. That way you can convert the fields at runtime and you don’t have to create another ADSO to convert the fields. Converting the fields at runtime saves space (but costs performance).

After you select an ADSO on the first screen, you’ll come across Figure 18

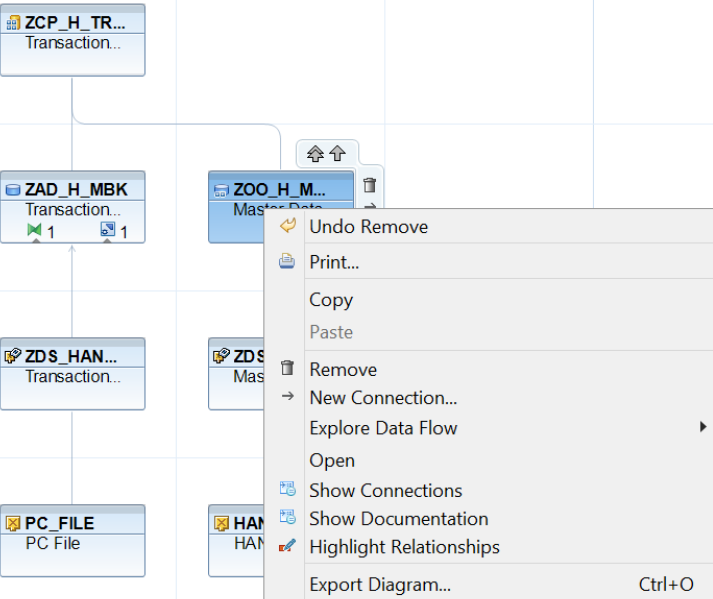


*Figure 18. Select a DataStore Object (advanced)*

You simply pick the ADSO you want, and press Finish to start modeling your OODS.

#### **5. Transformation**

After having read all of these Source Types you might be wondering - how do I transform my virtual data at runtime? After all, *if you right-click an Open ODS View you do not see any option to create a Transformation* (see Figure 19).

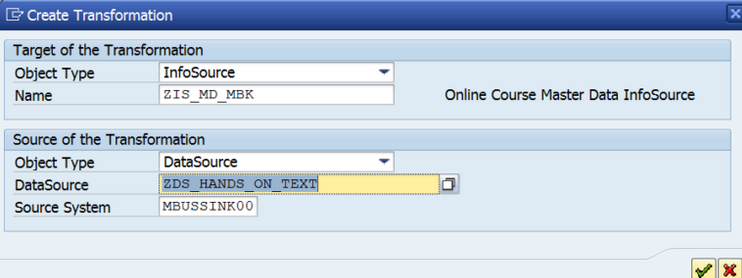


*Figure 19. No Transformation option available from the drop-down menu in BW/4HANA*

Well, I’m here to deliver you the good news - *you can transform your virtual data while you pull it in from an external source*. However, to do so you need 3 objects - a DataSource, an InfoSource and a Transformation. Keep in mind - *there will be a performance penalty, dependent on the complexity of the transformations you use*. Also, *you can neither use an OODS as a source or as a target of a Transformation in the Create Transformation screen*. You can only base an OODS on a Transformation by following the procedure I describe below.

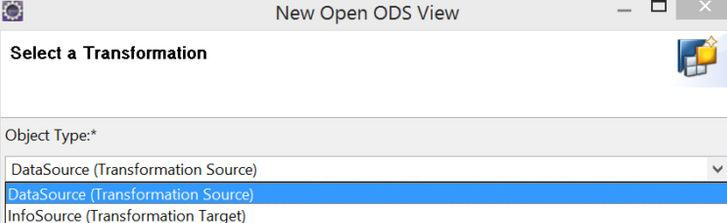
To create an OODS based on a Transformation take the following steps

1. If you haven’t done so already, first create a DataSource.
2. After that, create an InfoSource. By first creating a DataSource and then an InfoSource, you can use the DataSource as a template and save time.
3. *After you’ve created both objects, create a Transformation ( DataSource → InfoSource) by right-clicking the InfoSource and selecting the Create a Transformation option* (see Figure 20).



*Figure 20. Create Transformation screen*

1. *Create an OODS and select Transformation as Source Type.* On the following screen, you’ll be prompted to select a DataSource or InfoSource (seeFigure 21).



*Figure 21. Select a Transformation screen*

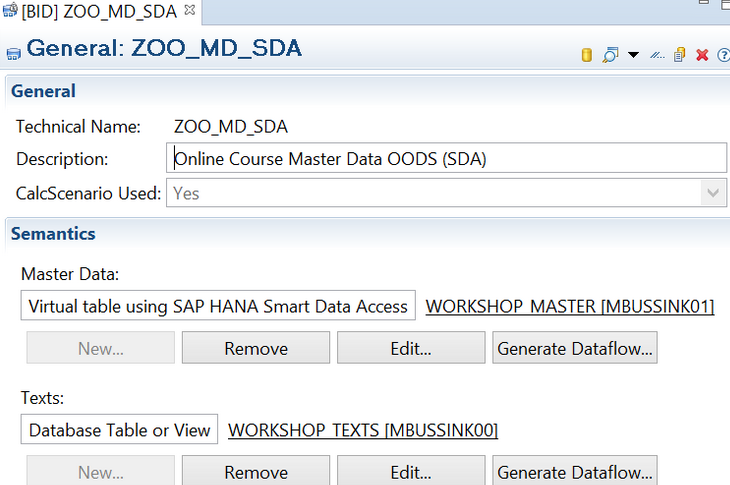
1. Select the DataSource you made during step 1, and the InfoSource you created during step 3. BW will automatically select the Transformation between your DataSource and your InfoSource.

Now you’ll be able to transform your data during runtime, when you execute a query based on your Open ODS View. However, I would only recommend using this Source Type for small datasets and for datasets with simple logic. *For more complicated logic and larger datasets, you’ll probably be better off transforming and persisting the data with an ADSO.*

### **General Tab**

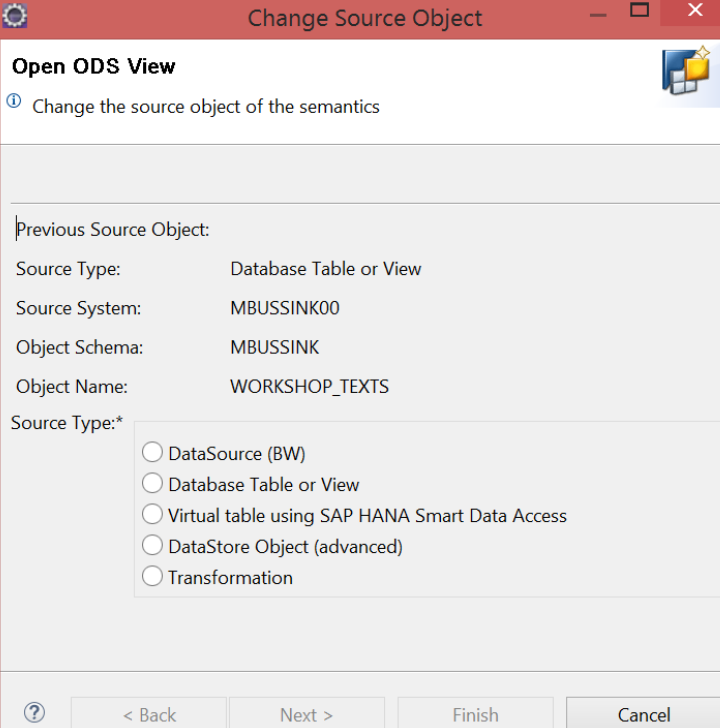
After you’ve picked a Source Type, you end up at the General Tab. Here *you can persist the OODS by clicking “Generate Dataflow”* (see Figure 22). *This option automatically creates an ADSO, Transformation and DTP for you.* Alternatively, it can also create an InfoSource, Transformation and DTP.

In the General Tab you can also change the OODS description, remove a Source and add a new one. If CalcScenario Used - yes, you can use the OODS in Join in a CompositeProvider, otherwise not. You can also change the source of the OODS by pressing “Edit” in the Semantics section of the General Tab.



*Figure 22. General tab Open ODS View*

After you click “Edit”, you get the following screen



*Figure 23. Change the Source object screen Open ODS View*

You can change the source of your OODS by following the steps and selecting a different source object.

### **Semantics**

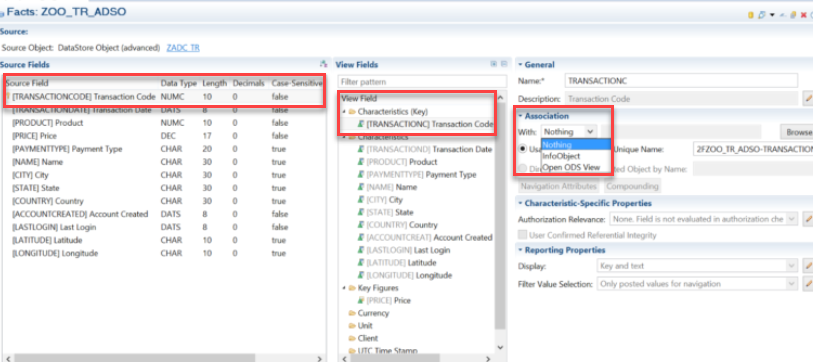
What are Open ODS semantics? Semantics let us categorize the data we use in the OODS. *You can categorize the Source data as either Facts, Master Data or Texts.* You can associate Fact-type Open ODS Views with Master Data and Text Open ODS Views. Association lets you use the Master Data OODS characteristics as Navigational Attributes and the Texts in the Text OODS for text handling, when you run a query on an Open ODS View or CompositeProvider. I demonstrate how association works in [Part II of this article](https://www.just-bi.nl/virtual-bw-hana-compositeprovider/).

Which type of fields, such as Characteristics, Timestamps and Valid From fields, are available in your OODS, depends on whether you define an OODS as a Fact, Master Data or Text OODS. *BW automatically maps the Source Fields to the View Fields, but you can change the mapping if it’s incorrect or incomplete*. You can also add the same field to multiple target fields, and give those extra target fields different descriptions and associations. This makes the OODS flexible. In the following sections, I’ll further explain the difference between the different Semantics.

#### **Facts**

When you select Facts in Figure 1, a tab called Facts appears in the modeling section of the OODS after you’ve completed the Source Type section. In Figure 24 we can see all the Source fields (on the left) and the View fields (in the middle). Adding fields to your View is easy due to the drag & drop-interface. You define your Source Fields as Characteristics, Key Figures, Currency etc. by dragging them to the corresponding folder in the View Fields section.  If you’re planning on persisting your OODS, it’s important to drag the key fields to the Characteristics (Key) folder. The fields you drag to this folder will turn into key fields in your generated ADSO.

Because we modeled this OODS without InfoObjects, each field gets a system-wide Unique Name. The system-wide Unique Name includes the InfoProvider in the name and the technical name of the field. For example, in Figure 24, our Transaction Code field has the name: 2FZOO\_TR\_ADSO-TRANSACTIONC.



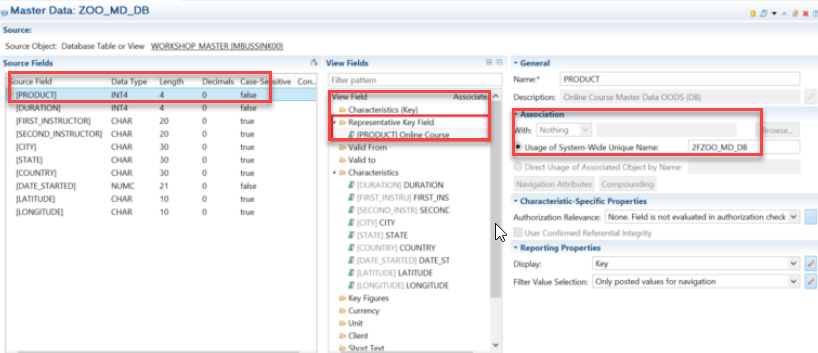
*Figure 24. Facts tab of Open ODS View*

Facts is the Semantic type you should use if you want to model a source table as transactional data. You can associate fields with Master Data/Texts Open ODS Views and InfoObjects in the Association area in the right panel. You thereby gain access to their metadata, authorization properties and Navigational Attributes (if you associate a characteristic field). Warning - *if you do not associate fields with InfoObjects, there can be a performance penalty*. This performance penalty is *due to the field structure having to be interpreted at query runtime.*

#### **Master Data**

When you choose Master Data during the OODS screen, you can use this OODS as a dimension/master data when modeling your HANA CompositeProvider (HCPR). How? By associating your fields with a Master Data OODS when you’re modeling a Fact OODS or a HCPR.

The Representative Key Field is the field you use to link your Master Data OODS to a Fact OODS or HCPR. Therefore, the field you drag to the Representative Key Field folder needs to be present in the Fact OODS /HCPR you want to link to your Master Data OODS. *By associating a field in a Fact OODS /HCPR with a Master Data OODS, you can use the rest of the Master Data OODS fields as Navigational Attributes.* Just don’t forget to drag them to the Characteristics folder in the View Fields section while modeling.



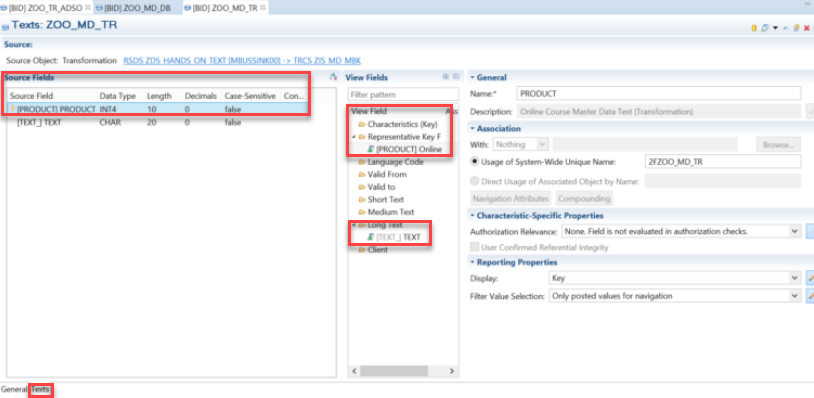
*Figure 25. Master Data Open ODS View*

In Figure 25 we can see that a Master Data Open ODS View contains more View Field types than a Facts Open ODS View. The additional fields are

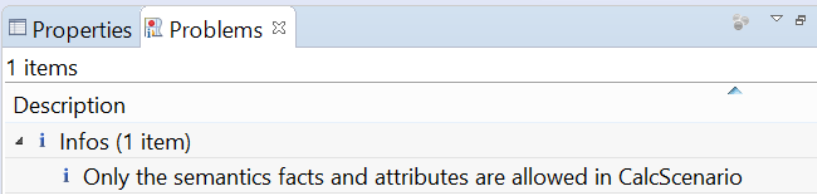
* **Representative Key Field** - this field *is used to link the Master Data OODS to a Fact OODS* or a CompositeProvider through Association. The Characteristics (Key) field in a Master Data OODS, is only used for additional compounded keys.
* **Valid From / Valid To** - *in case you have time-dependent data, you can drag the valid from and valid to fields to this folder*.
* Short / Medium / Long Text - drag any text fields one or more of these folders for text handling.

#### **Texts**

There is a 3rd type of Semantics - Texts (see Figure 26). You can already guess what it’s about - text tables (surprise, surprise). *Master Data Open ODS Views contain all the field semantics a Text Open ODS Views have, except for one - Language.* You might recognize Language as a standard field included in SAP Text tables.



*Figure 26. Texts Open ODS View*



*Figure 27. Warning during activation of Text Open ODS View*

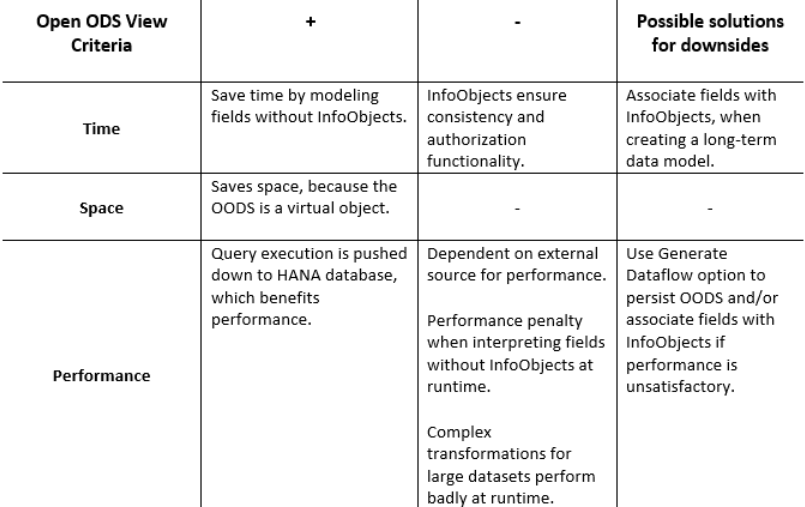
*You cannot use Text Open ODS Views in a CalcScenario* (see Figure 27). If CalcScenario - no, you cannot create a join using this OODS in a HCPR*. The only way to add a Text OODS View to a CompositeProvider is by* ***Association***, as I will explain next.

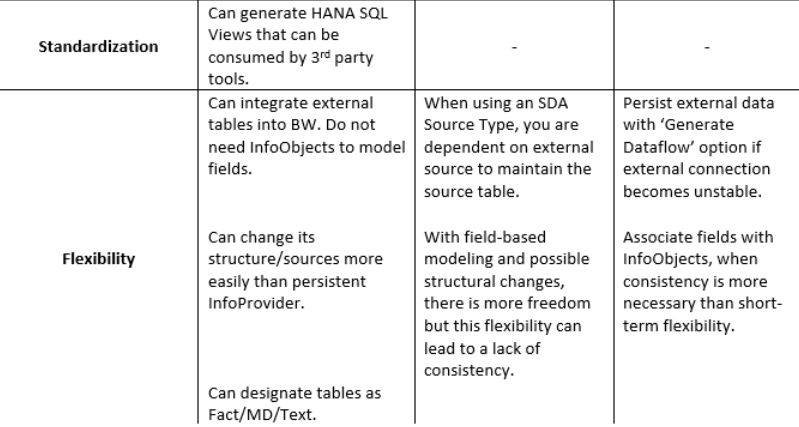
A Text OODS works very similarly to a Master Data OODS. *The field you use as Representative Key Field needs to be present in the Fact OODS or HCPR you want to link to the Text OODS. By associating that field with the Text OODS in the Fact OODS/HCPR*, you can use the text fields you defined in that Text OODS when executing a query. Don’t forget to drag the text fields to the Short/Medium/Long Text folders while modeling your Text OODS.

*You can also combine a Text OODS and a Master Data OODS.* You can combine them in the General tab of the OODS in the Semantics part (see Figure 22). To add a Text OODS to a Master Data OODS press “New” on the General Tab. After pressing “New” you can either add a Text OODS to a Master OODS or vice versa.

### **Pros & cons**

Finally, here’s an overview of the pros (преимущество) and cons (недостаток) of the Open ODS View



*Table 2. Open ODS View pros & cons*

### **The Road to Virtuality**

On the road to having a more virtual data warehouse, you can easily get lost. We’ve seen that Open ODS Views are very versatile. Therefore, to explore them properly you need to have patience and perhaps read a blog or two (wink (wɪŋk подмигивать), wink).

You now know all the different Source Types you can use in an Open ODS View. For instance, you now know how to transform data during runtime by basing your Open ODS View on a Transformation. Whether or not that’s sensible, I’ll leave up to you. We also created a Source System, to set up a connection between our Open ODS View and a local database table or an external virtual table. We also briefly discussed Association and the difference between the three Semantic Types, including when to use which one.

Armed with this knowledge you’re now almost ready for the Virtual BW world. There is one last stop on the road to a virtual BW - the HANA CompositeProvider. You can read Part II of this article [here](https://www.just-bi.nl/virtual-bw-hana-compositeprovider/). Bon voyage!