How to use ABAP CDS for Data Provisioning in BW

Applicable Releases:
SAP BW 7.5 SP5 powered by SAP HANA and BW/4HANA SP01

Author:
Anna Coniglio

Version 1.0
January 2017
All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary. The information in this document is proprietary to SAP. No part of this document may be reproduced, copied, or transmitted in any form or for any purpose without the express prior written permission of SAP AG.

This document is a preliminary version and not subject to your license agreement or any other agreement with SAP. This document contains only intended strategies, developments, and functionalities of the SAP product and is not intended to be binding upon SAP to any particular course of business, product strategy, and/or development. Please note that this document is subject to change and may be changed by SAP at any time without notice.

SAP assumes no responsibility for errors or omissions in this document. SAP does not warrant the accuracy or completeness of the information, text, graphics, links, or other items contained within this material. This document is provided without a warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, or non-infringement. SAP shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials. This limitation shall not apply in cases of intent or gross negligence.

The statutory liability for personal injury and defective products is not affected. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third-party Web pages nor provide any warranty whatsoever relating to third-party Web pages.

SAP “How-to” Guides are intended to simplify the product implementation. While specific product features and procedures typically are explained in a practical business context, it is not implied that those features and procedures are the only approach in solving a specific business problem using SAP NetWeaver. Should you wish to receive additional information, clarification or support, please refer to SAP Consulting.

Any software coding and/or code lines / strings (“Code”) included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, except if such damages were caused by SAP intentionally or grossly negligent.

Disclaimer
Some components of this product are based on Java™. Any code change in these components may cause unpredictable and severe malfunctions and is therefore expressively prohibited, as is any decompilation of these components.

Any Java™ Source Code delivered with this product is only to be used by SAP’s Support Services and may not be modified or altered in any way.
## Document History

<table>
<thead>
<tr>
<th>Document Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.10</td>
<td>&lt;&lt; Enter your summary of changes in this version &gt;&gt;</td>
</tr>
<tr>
<td>1.00</td>
<td>First official release of this guide</td>
</tr>
</tbody>
</table>
## Typographic Conventions

<table>
<thead>
<tr>
<th>Type Style</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Example Text</em></td>
<td>Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options. Cross-references to other documentation</td>
</tr>
<tr>
<td>Example text</td>
<td>Emphasized words or phrases in body text, graphic titles, and table titles</td>
</tr>
<tr>
<td>Example text</td>
<td>File and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools</td>
</tr>
<tr>
<td>Example text</td>
<td>User entry texts. These are words or characters that you enter in the system exactly as they appear in the documentation</td>
</tr>
<tr>
<td><code>&lt;Example text&gt;</code></td>
<td>Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system</td>
</tr>
<tr>
<td>EXAMPLE TEXT</td>
<td>Keys on the keyboard, for example, F2 or ENTER</td>
</tr>
</tbody>
</table>

## Icons

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Caution" /></td>
<td>Caution</td>
</tr>
<tr>
<td><img src="image" alt="Note or Important" /></td>
<td>Note or Important</td>
</tr>
<tr>
<td><img src="image" alt="Example" /></td>
<td>Example</td>
</tr>
<tr>
<td><img src="image" alt="Recommendation or Tip" /></td>
<td>Recommendation or Tip</td>
</tr>
</tbody>
</table>
# Table of Contents

1. Introduction .......................................................................................................................................... 4

2. Prerequisites .......................................................................................................................................... 5
   2.1 Minimum system Requirements ........................................................................................................ 5
   2.2 Creating the ODP Source system .................................................................................................... 5
   2.3 Application Development using ABAP CDS .................................................................................... 5

3. Background ........................................................................................................................................... 5
   3.1 Delta with deletion ............................................................................................................................ 5

4. Scenario 1 – Directly Access CDS-ODP source from BW ................................................................. 8
   4.1 Step 1 – Inspect CDS View .............................................................................................................. 9
   4.2 Step 2 – Create DataSource ............................................................................................................. 10
   4.3 Step 3 – Consume CDS view within BW modeling tools .............................................................. 13

5. Scenario 2 – Delta Extraction of CDS-ODP source to BW ............................................................... 18
   5.1 Step 1 – Create CDS View ............................................................................................................. 18
   5.2 Step 2 – Create BW DataSource .................................................................................................... 20
   5.3 Step 3 – Create BW Dataflow ....................................................................................................... 23
   5.4 Optional Step 4 – Activate Real-Time Streaming ........................................................................... 28

6. Replacing BW Extractors with CDS views in S/4HANA ..................................................................... 29

7. Appendix ............................................................................................................................................... 30
1. Introduction

With the availability of the SAP HANA platform there has been a paradigm shift in the way business applications are developed at SAP. The rule of thumb: *Do as much as you can in the database to get the best performance.*

To take advantage of the SAP HANA Database, SAP has introduced a new concept for data modelling called Core Data Service (CDS) which helps to facilitate the push-down of processes to the SAP HANA database. Data models based on CDS serve as central definitions that can be used in many different domains, such as transactional and analytical applications.

CDS is defined using an SQL-based data definition language (DDL) leveraging standard SQL with some additional concepts, such as associations, which define the relationships between CDS views, and annotations, which direct the domain-specific use of CDS artefacts. There are two types of CDS available, the HANA CDS (defined in XS engine) and ABAP CDS.

CDS has become the cornerstone for application development at SAP. A key example of this can be seen in S/4HANA which uses CDS to provide both core transactional functionality as well as analytical content for reporting.

From release BW 7.5 SP05 powered by HANA (BW/4HANA SP01), ABAP based CDS views can be used for data extraction. It is the purpose of this document to provide a step-by-step guide on how to configure this scenario. This paper will cover 2 use cases

- Direct access to a CDS view from BW and
- Delta extraction from a CDS view to BW

The following diagram describes these two scenarios
2. Prerequisites

2.1 Minimum system Requirements

Extracting data using a CDS view and the ODP framework requires a minimum version on both the source (ABAP) and target (BW) systems.

Source Requirements
Full Extraction or Direct Access scenarios require a minimum of NetWeaver 7.5 SP04 or equivalent. Delta Extraction scenarios require a minimum of NetWeaver 7.5 SP05 or SAP BW/4HANA SP01.

Target Requirements
All ODP-CDS Extraction scenarios require a minimum of NetWeaver 7.3 SP08 or equivalent (ODP recommendation).

2.2 Creating the ODP Source system

For the extraction of data into BW, ensure an ODP source system of type ABAP_CDS is available to connect to the source system. This may require configuration by the system administrator.

For further information on setting up the ODP Source System see the help documentation Creating an ODP Source System.

2.3 Application Development using ABAP CDS

CDS views in S/4HANA should be developed in line with the customer development standards and guidelines. Discussion into the correct usage of CDS views and annotations is beyond the scope of this documentation.

For further information see the help documentation See ABAP CDS Help
See ABAP CDS Development User Guide

3. Background

Data extraction is enabled within the CDS definition through a suitable Analytics Annotation.

The following annotations are relevant for data provisioning purposes and are applicable only from NetWeaver 7.5 SP05.

```json
Analytics : {
    dataExtraction : {
        enabled : boolean default true;
        delta : {
            byElement : {
                name : RefToElement;
                maxDelayInSeconds : Integer default 1800;
                detectDeletedRecords : boolean default true;
                ignoreDeletionAfterDays : Integer; }
        }
    }
}
```
How to integrate external data into SAP BW powered by SAP HANA

| Analytics.dataExtraction.enabled | Application developers can use this annotation to mark views that are suitable for data replication (for example, delta capabilities must be provided for mass data).

Scope: #VIEW
Evaluation Runtime (Engine): Specifies which view will be exposed in replication scenarios.

Note This view must be annotated with Analytics.dataCategory(except the value AGGREGATIONLEVEL).

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>The view is suitable for data replication. This is the default setting.</td>
</tr>
<tr>
<td>False</td>
<td>The view is not suitable for data replication.</td>
</tr>
</tbody>
</table>

| Analytics.dataExtraction.delta.byElement.name | Application developers can enable the generic delta extraction with this annotation. This is the element that should be used for filtering during generic delta extraction. This element can either be a date (ABAP type DATS) or a UTC time stamp.

Scope: #VIEW
Evaluation Runtime (Engine): Specifies which view will be exposed in replication scenarios.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>The view is suitable for data replication. This is the default setting.</td>
</tr>
</tbody>
</table>
How to integrate external data into SAP BW powered by SAP HANA

<table>
<thead>
<tr>
<th>Analytics.dataExtraction.delta.byElement.maxDelayInSeconds</th>
<th>There is always a time delay between taking a UTC time stamp and the database commit. This annotation specifies the maximum possible delay in seconds. Scope: #VIEW Evaluation Runtime (Engine): Specifies which view will be exposed in replication scenarios.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>Integer</td>
<td>Maximum number of seconds between taking the time stamp and the successful database commit. The default is 1800 seconds.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Analytics.dataExtraction.delta.byElement.detectDeletedRecords</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>Delta extraction captures deleted records</td>
<td></td>
</tr>
<tr>
<td>False</td>
<td>Delta extraction does not capture deleted records</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Analytics.dataExtraction.delta.byElement.ignoreDeletionAfterDays</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>Number of days delta queue captures deleted records.</td>
<td></td>
</tr>
</tbody>
</table>

Further information and a full list of analytical annotation go to the [Analytical Annotations Help Documentation](https://example.com).
3.1 Delta with deletion

The CDS analytics annotation supports delta extraction with deletion. When delete records are requested, an additional table is created, storing key fields which have already been extracted. During the extraction process the key fields of the source table and extracted records table are compared and delete records are generated.

4. Scenario 1 – Directly Access CDS-ODP source from BW

In this scenario data will be extracted from a standard CDS view in S/4HANA. The CDS view chosen represents Cost Center Activity Type master data, I_COSTCENTERACTIVITYTYPE. I_COSTCENTERACTIVITYTYPE is delivered as part of the Virtual Data Model (VDM) in S/4HANA Analytics and in this case already contains the data extraction annotation allowing the view to be used for data extraction. The reference to this annotation is highlighted below

```sql
define view I_CostCenterActivityType
as select from csia
```

Please note that because the delivered CDS view does not include the delta extraction annotation, only full extraction or direct access can be used in this scenario. Also keep in mind that not all VDM CDS views will contain the dataextraction:enabled annotation, this will only be used where extraction is possible and appropriate.

Scenario 1 will detail the process described below
### 4.1 Step 1 – Inspect CDS View

For this scenario, a pre-delivered CDS view will be used for data extraction. Although the object itself is already configured, some information from the CDS view is required and gathered in Step 1.

<table>
<thead>
<tr>
<th>4.1.1</th>
<th>In the HANA studio navigate to the S/4HANA project using the ABAP Development Perspective</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1.2</td>
<td>In the S/4HANA project, locate the CDS view which should be used for data extraction.</td>
</tr>
</tbody>
</table>

*In this case I_COSTCENTERACTIVITYTYPE will be used*

---

Note: If using a standard CDS View ensure the `dataExtraction.enabled: true` annotation is available. The dataExtraction annotation is used in standard VDM where data extraction from the view makes sense.
4.1.3 In the CDS view definition make the note of the CDS view name, Analytics data Category and SQL view name. This information will be used in subsequent steps

In this case,
The CDS view name = I_CostCenterActivityType
Data Category = DIMENSION
SQLViewName = IFICCActTyp

4.2 Step 2 – Create Datasource

4.2.1 Navigate to the BW/4HANA Project using the BW perspective

4.2.2 Under the DataSources folder find the ODP_CDS source system for the S/4HANA system

Note: As data extraction is client dependent, ensure the correct client for the system is selected – in this case 910 is used
4.2.3 Right click on the Source System object and select New -> DataSource

4.2.4 In the DataSource creation screen, select the browse button to use the Operational Data Provider as a template
4.2.5 In the browse screen use the SQLView name identified in the previous section to find the ODP source and click OK.

Note: You must use the SQL view name, not the CDS view name.

4.2.6 The DataSource input screen will be automatically filled with the information relating to the ODP source. Click Finish to continue the DataSource configuration.
4.2.7 The fields of the CDS view will be automatically included in the DataSource definition. Save and Activate the DataSource. Make note of the DataSource name for the next step in this scenario. 

In this case the DataSource name is IFICCACTTYP_P

The CDS view data can now be used within BW models and dataflows.

4.3 Step 3 – Consume CDS view within BW modeling tools

BW/4HANA offers a variety of data acquisition options. In this scenario direct access (through the ODP Framework) or a more traditional full data load scenario could be used. The following section will demonstrate the consumption of the data through direct access using an Open ODS View.

4.3.1 Navigate to the BW/4HANA system using the BW perspective
How to integrate external data into SAP BW powered by SAP HANA

| 4.3.2 | Right click on the BW/4HANA project and select New -> Open ODS View |

![Image](image_url)
4.3.3 In the Open ODS View creation screen fill details for:
- InfoArea,
- Technical name
- Description

Choose the Semantic type for the Open ODS View.

In this case the CDS view in the S/4HANA system has been defined with the DIMENSION annotation (identified in Step 1), so this Open ODS will be for Master Data.

Note: CDS views with analytics annotation type AGGREGATIONLEVEL cannot be used for data extraction.

Select the source type. This example will use direct access from a DataSource so DataSource(BW) is selected.

Click Next.
4.3.4 Search for the BW Datasource name (noted in the previous steps). Select and click next. In this case the DataSource name is IFICCACTTYP_P

4.3.5 BW/4HANA will automatically make a proposal for the fields in the Open ODS view definition. Save and Activate.
4.3.6 To view the data, use the Data Preview button.

The open ODS view will provide direct access to the CDS view via the BW DataSource.
5. **Scenario 2 – Delta Extraction of CDS-ODP source to BW**

Scenario 2 will demonstrate data extraction from a custom CDS view in S/4HANA using the Delta mechanism. In this case, GL Line Item information will be extracted, sourcing data primarily from the table ACDOCA in S/4HANA. The ACDOCA table is typically a very large table, so extracting all the records regularly (as full loads) or using direct access to BW would not be recommended. Instead, data extraction using the delta mechanism is preferred whereby only ‘New’, ‘Changed’ or ‘Deleted’ records are extracted periodically. CDS views allow delta extraction via the data extraction and delta element annotations. These annotations require a delta element or a ‘pseudo delta’ field to be defined – typically a last changed timestamp on the record.

Not all tables/views in S/4HANA will contain the last changed timestamp and may need to be derived by joining CDS views. In this case the standard model is enhanced with data from table BWFI_AEDAT which supplies the last changed at timestamp.

The following diagram describes the steps of this scenario.

### 5.1 Step 1 – Create CDS View

| 5.1.1 | In the HANA studio navigate to the S/4HANA project using the ABAP Development Perspective |
### 5.1.2 Right click on the Project and select New -> Other

![Image](image1.png)

### 5.1.3 From the Wizard, navigate to ABAP- >Core Data Services -> Data Definition and click Next

![Image](image2.png)

### 5.1.4 Choose the Package, Technical Name and Description. In this case the technical name is ZI_BW_GLACCTLINE

![Image](image3.png)

### 5.1.5 Enter the CDS view code. The full code is available in the appendix of this document.

![Image](image4.png)
5.2 Step 2 – Create BW DataSource

5.2.1 Navigate to the BW/4HANA system using the BW perspective

5.2.2 In the DataSources folder, find the ABAP CDS source system for the S/4HANA system. Right click and select New -> Datasource
5.2.3 Click the Browse button to use an ODP source as a template

5.2.4 Find the CDS View and click OK
5.2.5 The DataSource input fields will be automatically be filled. Click Finish.

5.26 The fields of the CDS view will be automatically included in the DataSource definition. Save and Activate the DataSource. Make note of the DataSource name for the next step in this scenario.

*In this case the DataSource name is IFICCACCRTYP_P*
The CDS View can now be used within BW models and dataflows. A simple data flow will be created to load the data into BW.

5.3 Step 3 – Create BW Dataflow

5.3.1 Locate the DataSource created above.
Right Click and select ‘Explore DataFlow’
The dataflow is very simple containing only the Source System object and Datasource itself.

5.3.2 Drag the aDSO object to the Dataflow working area

5.3.2 Right click on the aDSO object (which has not been created yet) and select New Connection
<table>
<thead>
<tr>
<th>5.3.3</th>
<th>Select the DataSource and click OK</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>5.3.4</th>
<th>Right click the aDSO object and select Create Persistent BW object</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image2.png" alt="Image" /></td>
<td></td>
</tr>
</tbody>
</table>
5.3.5 Fill out the technical Name and description and click Finish

5.3.6 All the fields of the DataSource have automatically been copied to the aDSO definition. Save and Activate

5.3.7 In the Dataflow definition, right click on the aDSO object, click Transformation -> Create Transformation
5.3.8 BW/4HANA will automatically provide suggestions within the transformation definition.
Save and Activate

5.3.9 Right click on the aDSO definition and select DTP’s-> Create DTP

5.3.10 Select the transformation and click OK
5.3.11 In the extraction screen, notice that both Delta and Full load options are available. In this case select Delta. Activate DTP
Note: In the DTP definition, Filter values can also be selected

5.3.12 To Load Data to the aDSO, go to the Execute tab and click Execute

5.3.13 In this step the data will be extracted from S/4HANA to BW/4HANA. Go to the monitor to check the BW Load.
5.3.14 The loads and delta can be monitored and managed in the Source system (S/4HANA) through transaction ODQM0N

5.4 Optional Step 4 – Activate Real-Time Streaming
Real-time streaming can be enabled for the dataflow created in the previous step.

5.4.1 In the HANA Studio, locate the BW project and select ‘Open SAP GUI’ button

5.4.2 Go to transaction RSPC

5.4.3 Click the Create new button
Enter Process Chain Technical Name and Description

5.4.4 Enter the Start Process technical name and description
| How to integrate external data into SAP BW powered by SAP HANA |

**5.4.5**
Add the DTP and aDSO activation processes to the Process Chain

![Diagram of Data Transfer Process and DSO (advanced)]

**5.4.6**
Select the streaming properties button from the menu and select the ‘Process Chain Runs in Streaming Mode’ check box. Set the Error threshold and Log retention according to requirements. Click OK

![Streaming Property Window](Streaming.png)

**5.4.6**
Save and activate the process chain
The process chain is now streaming.

---

### 6. Replacing BW Extractors with CDS views in S/4HANA

For the most-part, traditional BW extractors can be used to extract data from an S/4HANA system. Due to simplifications in the S/4HANA system, some extractors have been deprecated or can no longer be used in the same way. For these extractors, SAP may deliver CDS Views which are extraction relevant (via the analytical annotations). In this case, the methods of extracting and consuming data described in this document can be used.
7. Appendix

@AbapCatalog.sqlViewName: 'ZIBWGLLINE2'
@AbapCatalog.compiler.compareFilter: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'GL Account Line Item (BW Extraction)'
@Analytics: {
  dataCategory: #FACT,
  dataExtraction: {
    enabled: true,
    delta.byElement.name: 'lastChangedAt',
    delta.byElement.maxDelayInSeconds: '100'}
}

define view Zi_Bw_Glacctline2 as select from I_GLAccountLineItem
association [1..*] to Zi_Bwfi_Aedat as _BWDELTA
  on I_GLAccountLineItem.CompanyCode = _BWDELTA.CompanyCode and
  I_GLAccountLineItem.AccountingDocument = _BWDELTA.AccountingDocument and
  I_GLAccountLineItem.FiscalYear = _BWDELTA.FiscalYear
{
  key I_GLAccountLineItem.CompanyCode,
  key I_GLAccountLineItem.FiscalYear,
  key I_GLAccountLineItem.AccountingDocument,
  key I_GLAccountLineItem.LedgerGLLineItem,
  I_GLAccountLineItem.Ledger,
  I_GLAccountLineItem.SourceLedger,
  I_GLAccountLineItem.LedgerFiscalYear,
  I_GLAccountLineItem.ChartOfAccounts,
  I_GLAccountLineItem.ControllingArea,
  I_GLAccountLineItem.GLAccount,
  I_GLAccountLineItem.CostCenter,
  I_GLAccountLineItem.ProfitCenter,
  I_GLAccountLineItem.FunctionalArea,
  I_GLAccountLineItem.BusinessArea,
  I_GLAccountLineItem.Segment,
  I_GLAccountLineItem.BalanceTransactionCurrency,
  I_GLAccountLineItem.AmountInBalanceTransacCrcy,
  I_GLAccountLineItem.BaseUnit,
  I_GLAccountLineItem.Quantity,
  I_GLAccountLineItem.FixedQuantity,
  I_GLAccountLineItem.CostSourceUnit,

  /* Associations */
  _BWDELTA.ChangedTimeStamp,

  @Semantics.systemDate.lastChangedAt

  case when _BWDELTA.ChangedTimeStamp > I_GLAccountLineItem.CreationDateTime
    then cast(cast(cast(substring(cast(_BWDELTA.ChangedTimeStamp as abap.char(25)),
      1, 14) as abap.numc(14)) as abap.dec(15, 0)) as rvlsttstmp)
    else cast(cast(cast(substring(cast(I_GLAccountLineItem.CreationDateTime as
      abap.char(25)), 1, 14) as abap.numc(14)) as abap.dec(15, 0)) as rvlsttstmp)
  end as lastChangedAt
}