

PUBLIC
SAP BW/4HANA
2024-02-23

SAP BW/4HANA 2023 SPS00 Modeling User Guide

© 2024 SAP SE or an SAP affiliate company. All rights reserved.

THE BEST RUN SAP

Content

1 Modeling with the BW Modeling Tools in Eclipse. 10

2 Getting Started: Working with BW Modeling Tools. 11

2.1 Basic Concepts. 11

"BW Modeling" Perspective. 11

Editors. 12

Views. 12

2.2 Basic Tasks. 18

Working with BW Projects. 18

Working with BW Metadata Objects. 23

2.3 Reference. 35

Keyboard Shortcuts for BW Modeling Tools. 35

Display Options for the BW Modeling Tools. 36

Settings for BW Modeling Tools. 37

3 Concepts. 39

3.1 Modeling with Fields Instead of InfoObjects. 39

3.2 InfoArea. 40

Transporting InfoAreas. 40

3.3 Data Flow. 41

Data Flow Object. 42

Documenting Data Flows. 44

Transient Data Flow. 44

Transporting Data Flows. 45

3.4 Open ODS View. 45

Components of the Open ODS View Definition. 46

Supported Sources for Open ODS Views. 47

Suppress Read Access for Unavailable Sources Temporarily 49

Structure of the Open ODS View Editor. 49

InfoObject Names for Fields in Open ODS Views. 51

Authorization Behaviour for Fields of Open ODS Views. 51

Transporting Open ODS Views. 53

Transformation of Data from External Sources for Open ODS Views 54

3.5 InfoObject. 54

Characteristic. 54

Key Figures. 63

Reference InfoObjects. 64

Unit. . . . 64
XXL InfoObject. . . . 65
Authorizations for InfoObjects. . . . 65
Transporting InfoObjects. . . . 65
3.6 InfoSource. . . . 65
Recommendations for Using InfoSources. . . . 67
Authorizations for InfoSources. . . . 69
Transporting InfoSources. . . . 69
3.7 DataStore Object. . . . 70
Structure of the DataStore Object 70
Modeling Properties. . . . 71
Data Tiering Properties. . . . 81
Currency Translation and Unit Conversion. . . . 82
APIs for the DataStore Object. . . . 83
Using the Concept of Non-Active Data. . . . 83
External SAP HANA SQL View. . . . 88
Transporting DataStore Objects. . . . 91
3.8 Semantic Groups. . . . 91
Authorizations for Semantic Groups. . . . 92
Transporting Semantic Groups. . . . 92
3.9 CompositeProvider. . . . 92
Supported Providers. . . . 93
Using HDI Containers. . . . 94
Supported Assignments. . . . 95
Temporal Join. . . . 95
Possible Assignments of Data Types. . . . 95
Scenarios for Using CompositeProviders in CompositeProviders. . . . 98
Query Properties. . . . 101
Audit. . . . 102
Names for CompositeProvider Fields in the Query. . . . 108
Authorizations for CompositeProviders. . . . 109
Transporting CompositeProviders. . . . 109
Recommendations for Modeling CompositeProviders. . . . 109
3.10 Document in the Document Store. . . . 110
3.11 Query. . . . 111
Characteristic. . . . 112
Structure. . . . 112
Structure Element. . . . 113
Selection. . . . 114
Formula. . . . 114
Cell, Exception Cell and Help Cell. . . . 114

Filter. . . . 115
Condition. . . . 116
Exception. . . . 117
Reusable Query Components. . . . 117
Linked Component. . . . 140
FEMS. . . . 141
Transporting Queries. . . . 141
3.12 Source System. . . . 142
Transferring Data with an SAP HANA Source System. . . . 142
Transferring Data with a Big Data Source System. . . . 161
Transferring Data Using Operational Data Provisioning. . . . 161
Transferring Data from Flat Files. . . . 185
Application Component Hierarchy. . . . 185
3.13 DataSource. . . . 187
3.14 Transformation. . . . 187
Rule Group. . . . 188
Semantic Grouping. . . . 189
Error Analysis in Transformations. . . . 190
Authorizations for the Transformation. . . . 190
Transport the Transformation. . . . 190
3.15 Data Transfer Process. . . . 190
Data Transfer Intermediate Storage (DTIS). . . . 191
Handling Data Records with Errors. . . . 192

4 Tasks. . . . 195
4.1 Creating InfoAreas. . . . 195
Moving InfoAreas and Objects in the InfoArea Hierarchy. . . . 196
Deleting InfoAreas. . . . 196
4.2 Working with Data Flows. . . . 197
Creating a Data Flow Object. . . . 197
Creating Documentation. . . . 208
Saving Data Flow Documentation as an HTML File. . . . 209
Copying a Data Flow. . . . 210
Using a Transient Data Flow for a Persistent Object. . . . 213
Tips and Tricks for Working with Data Flows. . . . 214
Recommendations for Working with Large Data Flows. . . . 214
4.3 Creating an Open ODS View. . . . 215
Creating an Open ODS View Using the Wizard (Defining Semantics and Source). . . . 216
Specifying the Structure of the Open ODS View. . . . 224
Displaying Initial Properties for Open ODS View Fields. . . . 225
Changing Properties for Fields of Type "Characteristic". . . . 228
Changing Properties for Fields of Type "Key Figure". . . . 231

Adding Texts or Attributes. . . . 232
Replacing the Source of an Open ODS View. . . . 233
Creating a Data Flow for Open ODS Views. . . . 234
4.4 Creating InfoObjects. . . . 237
Creating Characteristics. . . . 238
Creating Key Figures. . . . 265
Creating a Unit. . . . 269
Creating XXL InfoObjects. . . . 269
Creating a Copy of an InfoObject. . . . 270
4.5 Creating InfoSources. . . . 270
Creating InfoSources Using the Wizard. . . . 271
Adding InfoObjects. . . . 272
Adding Fields. . . . 272
4.6 Creating DataStore Objects. . . . 273
Creating DataStore Objects Using the Wizard. . . . 274
Adding InfoObjects. . . . 275
Adding Fields. . . . 276
Defining Keys. . . . 277
Using Non-Cumulative Key Figures. . . . 277
Creating Partitions. . . . 278
Creating Indexes. . . . 283
Define Anonymization Rules. . . . 283
Technical Settings. . . . 286
4.7 Creating Semantic Groups. . . . 288
Example of the Definition of a Semantic Group. . . . 290
Changing the Semantic Group and Its Objects. . . . 292
Creating a Copy of a Semantic Group. . . . 293
Deleting Semantic Groups. . . . 294
Managing Semantic Group via API. . . . 294
4.8 Creating CompositeProviders. . . . 295
Creating CompositeProviders Using the Wizard. . . . 297
Using a CompositeProvider as a Provider in Other CompositeProviders. . . . 297
Creating Assignments. . . . 298
SQL Expression. . . . 300
Using Temporal Joins. . . . 301
Editing Groups. . . . 302
Editing Properties. . . . 303
Changing Properties for Fields of Type "Characteristic". . . . 304
Activating Navigation Attributes. . . . 306
Changing Properties for Fields of Type "Key Figure". . . . 307
Changes to Used Objects. . . . 309

Creating a Copy of a CompositeProvider. 310
Supplying Data to 0INFOPROV. 310
Processing Joins. 311
Settings for CompositeProviders. 311
Pruning Queries on CompositeProviders. 311
4.9 Creating a Document Store. 314
4.10 Working with Queries. 316
Creating a Query Using the Wizard. 317
Defining Queries in the Editor. 318
Opening and Changing Existing Queries. 444
4.11 Creating a Link Component. 445
Apply Changes from the Source Component. 446
Changing a Target Component. 447
4.12 Using the Open Hub Destination to Distribute Data from the SAP BW/4HANA System. 448
Creating Open Hub Destinations. 449
Selecting Database Tables as Destinations. 450
Selecting Files as Destinations. 453
Edit Fields. 454
4.13 Working with Source Systems. 455
Creating a Source System. 455
Hiding Empty Application Components. 466
Creating and Moving Application Components in a Generic Hierarchy. 467
Executing Replication. 468
Set Source System to Inactive. 469
4.14 Working with DataSources. 470
Creating DataSources. 471
Editing DataSources. 496
Metadata Comparison and Updating DataSources. 497
4.15 Creating Transformations. 497
Differences Between ABAP Runtime and SAP HANA Runtime. 500
Creating a Transformation Using the Wizard. 501
Create Rule Group. 501
Transformation in the SAP HANA Database. 502
Editing Rule Details. 503
ABAP Routines in Transformations. 511
SAP HANA SQLScript Routines in the Transformation. 533
4.16 Creating a Data Transfer Process. 535
Creating a Data Transformation Process Using the Wizard. 536
Editing General Properties. 537
Editing Extraction Properties. 539
Editing Update Properties. 540

Editing Runtime Properties. . . . 544
Excuting Data Transfer Process. . . . 545
Monitoring Data Transfer Process. . . . 547
Displaying Properties of the Data Transfer Process. . . . 548

5 Security Aspects for the Modeling Tools in Eclipse. . . . 550
5.1 User Authentication on the Front End Client. . . . 550
5.2 Authorizations for Modeling with the Modeling Tools. . . . 551
5.3 Protecting Resources on the Front End Client. . . . 553
5.4 Installing Third-Party Plug-Ins. . . . 554

6 What's New in the BW Modeling Tools. . . . 555
6.1 Version 1.25. . . . 557
Merge Mode for DataStore Objects (New). . . . 557
Mass Activation (New). . . . 558
Transport Overview (New). . . . 558
Enhanced Master Data Update - Mixed Case (Enhanced). . . . 558
InfoObject with Write Interface (New). . . . 558
6.2 Version 1.23. . . . 559
Content Explorer View and Content Install View (New). . . . 559
Operations in SAP HANA (Enhanced). . . . 559
DataStore Object - Dynamic Partitioning (Enhanced). . . . 560
DataStore Object - Technical Settings (Enhanced). . . . 560
Data Tiering Optimization - Exceptional Updates (New). . . . 560
6.3 Version 1.22. . . . 561
Query: Restrict Formula (New). . . . 561
Query: Precalculate Value Set (New). . . . 561
6.4 Version 1.21. . . . 561
DataStore Object (Changed). . . . 562
External SAP HANA SQL View for DataStore Objects (New). . . . 562
InfoObject (Enhanced). . . . 562
Data Flow Modeling (Enhanced). . . . 563
Enhanced Data Flow Copy (New). . . . 563
Grouping of Process Chains into InfoAreas (New). . . . 564
Variable with Replacement from BRFplus (New). . . . 564
Operations in the SAP HANA (Enhanced). . . . 564
Currency Translation and Unit Conversion (Changed). . . . 565
Key Date Derivation Type (Changed). . . . 565
BW Job Repository Feed (New). . . . 565
BW Job View (New). . . . 565
BW Modeling Tools - Miscellaneous (Changed). . . . 566
6.5 Version 1.19. . . . 566

Transformation (changed). . . . 567
Data Transfer Process (Changed). . . . 567
Optimizer for Operations in SAP HANA (New). . . . 568
Editor for Document Store (New). . . . 569
Open ODS View-Wizard and Editor (Changed). . . . 569
Wizard for Creating Source Systems (Changed). . . . 570
Editor for File DataSource (New). . . . 570
DataSource-Editor (Enhanced). . . . 570
Open Hub Destination (Enhanced). . . . 571
Enhanced Master Data Update (changed). . . . 572
Authorizations (Changed). . . . 572
Version Comparison (New). . . . 572
6.6 Version 1.18. . . . 573
Query: Assignment to Role (New). . . . 573
Query: Linked Component (New). . . . 573
InfoProvider Views in the BW Modeling Tools (Enhanced). . . . 574
Automatic Object Check During Processing in the BW Modeling Tools (New). . . . 574
Wizard for Creating DataSources (Changed). . . . 574
Wizard for Creating Source Systems (Changed). . . . 575
6.7 Version 1.17. . . . 575
Creating Source Systems in BW Modeling Tools (new). . . . 575
6.8 Version 1.16. . . . 576
Navigation Attributes in CompositeProviders (Changed). . . . 576
Big Data Source System (New). . . . 576
Data Transfer Process and Transformations in Project Explorer View (New). . . . 577
Creating a Query: Defining Priorities (New). . . . 577
6.9 Version 1.15. . . . 577
Data Flow (new). . . . 577

7 More Information. . . . 579
7.1 Rules for Standard Aggregation and Exception Aggregation. . . . 579
7.2 Formula Calculation. . . . 583
Basic Functions. . . . 584
Percentage Functions. . . . 584
Data Functions. . . . 587
Mathematical Functions. . . . 598
Trigonometric functions. . . . 600
Boolean Operators. . . . 601
7.3 Using Precalculated Value Sets (Buckets). . . . 605
7.4 CURRENT MEMBER Variables. . . . 607
Example Query for Using a CURRENT MEMBER Variable. . . . 611

Examples of Using the CMR Operator. 612

1 Modeling with the BW Modeling Tools in Eclipse

The *BW Modeling tools* provide an Eclipse-based integrated modeling environment for the management and maintenance of SAP BW/4HANA metadata objects.

The main objective of this is to support metadata modelers in today's increasingly complex BI environments by offering flexible, efficient and state-of-the-art modeling tools. These tools integrate with ABAP Development Tools as well as with SAP HANA modeling and the consumption of SAP HANA elements in metadata objects, like Open ODS views or CompositeProviders. The BW Modeling Tools have powerful UI (user interface) capabilities.

This documentation provides information about how you can set up projects in order to perform BW modeling tasks. It also describes how to work with SAP BW/4HANA metadata objects in the Eclipse-based integrated development environment (IDE). In particular, it describes how to define SAP BW/4HANA metadata objects (like DataStore objects, CompositeProviders or queries) using native Eclipse-based editors.

Validity of the Documentation

This documentation belongs to **Version 1.22** of the BW Modeling Tools and refers to the functions that are part of the standard delivery for SAP BW/4HANA 2.0 SPS04.

2 Getting Started: Working with BW Modeling Tools

The BW Modeling tools provide you with the *BW Modeling* perspective, with predefined views and configuration. This perspective provides you with the tools and functions required for modeling tasks. You organize your tasks in projects.

The following sections provide information about the basic concepts and tasks in the *BW Modeling* perspective, with special focus on organizing and structuring work in projects, and the general functions for working with BW metadata objects.

2.1 Basic Concepts

The following sections introduce the basic concepts for working with the BW Modeling tools.

2.1.1 "BW Modeling" Perspective

Just like any other perspectives in Eclipse, the *BW Modeling* perspective defines the initial set and layout of tools (views and editors) in the Eclipse window. It thus provides a set of functions aimed at accomplishing modeling tasks in SAP BW/4HANA.

In particular, it enables working with SAP BW/4HANA metadata objects that are managed by a back end system. When using the *BW Modeling* perspective, you always have to establish a system connection that is managed by a corresponding BW project. You can use the BW project to access the system's metadata objects.

The BW perspective enables access to both Eclipse-based and GUI-based modeling editors. It consists of an editor area, where the SAP BW/4HANA metadata object editors are placed, and the following views:

- Project Explorer
- Properties
- Problems
- History
- BW Reporting Preview
- InfoProvider
- Outline

Related Information

Reference [page 35]

2.1.2 Editors

The Editor area provides one or more editors for SAP BW/4HANA metadata objects.

> **Note**
>
> When opening a native Eclipse editor for a SAP BW/4HANA metadata object (a CompositeProvider or query for example), the object is not locked. The object will only be locked once the user starts working with it. A message to this effect then appears in the header of the BW modeling tools.

2.1.3 Views

The following sections describe the default views in the *BW Modeling* perspective.

2.1.3.1 Project Explorer View

Projects are the largest structural unit used by the Eclipse Workbench. They are displayed in the *Project Explorer* view. A BW project manages the connection to a back end system in the Eclipse-based IDE. It provides a framework for accessing and modeling SAP BW/4HANA metadata objects and functions in the Eclipse-based IDE.

A BW project always represents a real system logon to one single back end system and also offers a user-specific view of metadata objects (for example, DataSources, InfoProviders, InfoObjects or queries) contained in the system.

The specification for a BW project contains the following items:

- Project name
- System data – including system ID, client, user and password
- Default language - project-specific preference for predefining the original language of the objects that are created

As with all other project types in Eclipse, BW projects also form part of a user-specific workspace and thus define the starting point for all modeling activities in the Eclipse-based IDE. Several BW projects can be contained in a workspace. This means that the modeler can work on multiple back end systems in one and the same IDE session, without having to leave the immediate work environment. A modeler can also structure and organize the modeling activities in multiple BW projects connecting to one and the same back end system.

BW projects can only be processed if there is a system connection (to the back end). It is therefore not possible to have read or write access to the content of a BW project in offline mode.

BW projects and their structures are displayed in the *Project Explorer* view together with other projects of different project types (ABAP projects for example).

The structure of a BW project contains a list of SAP BW/4HANA metadata objects that are itemized under one of the following nodes:

- *Favorites* – Favorites that are stored in the associated back end system
- *BW Repository* – metadata objects like InfoProviders, InfoObjects and queries that are sorted according to InfoAreas or semantic folders for object types
- *DataSources* – DataSources that are sorted according to source system type, source system and application component

You can also assign the underlying SAP HANA database of the system, thus enabling consumption of SAP HANA views (SAP HANA information models) in SAP BW/4HANA metadata objects. The BW project structure then contains a further node, *SAP HANA System Library*, which lists the SAP HANA views.

2.1.3.1.1 Favorites

In your BW project, you will generally work with a limited selection of InfoAreas and metadata objects. You can assign objects that are especially relevant for your work to the *Favorites* node. The favorites are stored in the system and are available to you in all projects with a connection to the system.

The *Favorites* node in a BW project represents a user-defined and structured collection of InfoAreas and metadata objects, stored for all projects in the connected system.

The *Favorites* node is one of the nodes directly below the root of the project tree. Under this node, the InfoAreas ( ) assigned to the favorites are listed, together with their metadata objects. The metadata objects assigned individually to the favourites in the corresponding semantic folder ( ), for CompositeProviders or DataStore objects for example, are also listed.

Related Information

Managing Favorites [page 29]

2.1.3.1.2 BW Repository

The *BW Repository* node in a BW project represents the InfoArea tree of the connected system.

Related Information

Creating InfoAreas [page 195]

2.1.3.1.3 DataSources

The *DataSources* node in a BW project displays the overview of the source systems and DataSources of the connected system.

In the *Data Sources* tree, folders are displayed for the source system types. Under the source system types, the system displays the existing source systems with their application component hierarchy and the DataSources.

Related Information

Source System [page 142]
Working with Source Systems [page 455]

2.1.3.2 Properties View

The *Properties* view displays metadata, like package, version, responsible or last modifier for the SAP BW/4HANA metadata object shown in the active editor. Further information is displayed depending on the object, such as DDIC information for DataStore objects or technical attributes for DataSources.

Whenever you change the active editor, the *Properties* view automatically switches to the metadata information of the metadata object in the active editor.

→ Tip

- By clicking on the package, you can open an application displaying the package information.
- With the toggle button in the *Properties* view toolbar, you can pin the view to the currently selected editor. If you use this function, the *Properties* view will not be updated when you change the active editor.

2.1.3.3 Problems View

The *Problems* view displays messages pertaining to operations (for example checks) executed on SAP BW/4HANA metadata objects. The messages are displayed regardless of the editor that is currently active.

→ Tip

- By choosing *Full Description* in the context menu of a message, you can display detailed information about the message.
- In the *Configure Contents* dialog box in the *Problems* view, metadata object-related message types can be selected. Message types that are deselected will not be shown in the *Problems* view.

2.1.3.4 History View

The *History* view displays a list containing the current and historical versions of the SAP BW/4HANA metadata object and thus allows you to keep track of changes made to the object over time.

Whenever the object is saved, the system creates a modified version (M version). Whenever the object is activated, the system creates an active version (A version). If there is an A version already prior to activation, a historical version of the existing A version is also created upon activation. The historical versions provide a snapshot of the status of the object at a given point in time, and are numbered consecutively.

You can choose from the following functions:

- You can open a historical version from the *History* view in display mode.
- You can compare two versions with one another.

→ Tip

- You can call the *History* view by pressing *Show in history view* in the toolbar in an Eclipse-based editor.
- With the *Link with Editor and Selection* switch in the toolbar for the view, you can link the view with the currently active editor. The *History* view is then updated each time you switch the active editor, hence each time you display a different metadata object in the editor. In this case, the *History* view always shows the versions for the object that is displayed in the active editor. If the switch is not set, the *History* view is not updated when you display a different metadata object in the editor. Using the *Link with Editor and Selection* switch means that is not possible to use the *Pin this History View* switch.
- Using the *Pin this History View* switch in the toolbar for the *History* view, you can fix the *History* view to the currently selected editor (to the metadata object currently displayed). If you use this function, the *History* view is not updated when you display a different metadata object in the editor. Using the *Pin this History View* switch means that is not possible to use the *Link With Editor and Selection* switch.
- If you choose *Open* in the context menu of an entry in the *History* view, the editor is opened and displays the corresponding version of the object.

2.1.3.5 BW Reporting Preview View

The *BW Reporting Preview* view displays a reporting preview for the SAP BW/4HANA metadata object in the active editor.

→ Tip

You can open the reporting preview by doing one of the following:

- Choosing *Refresh Data Preview* on the *BW Reporting Preview* view toolbar
- Choosing *Show Preview* on the toolbar of an Eclipse-based editor

2.1.3.6 InfoProvider View

The *InfoProvider* view displays the SAP BW/4HANA metadata object in the active editor. This is based on how the InfoProvider is displayed in query maintenance. Ther InfoObjects are categorized here by *Key Figures*, *Characteristics* and *Reusable Components*.

> → Tip
>
> For supported objects, you can call the *InfoProvider* view by pressing *Show InfoProvider View* in the toolbar in an Eclipse-based editor.

You can choose from the following functions in the InfoProvider view:

- From the context menu on an InfoObject and reusable query component (such as a calculated key figure), you can open the object or run a where-used list for it.
- From the context menu for an InfoObject, you can define variables.
- From the context menu on the *Reusable Components* folder, you can create calculated and restricted key figures and global structures and filters.
- From the context menu on a folder for a certain type of reusable query component, such as *Restricted Key Figures*, you can create a reusable query component of the same type.
- From the context menu of a folder in the directory structure for calculated and restricted key figures, you can create further folders. You can thus create a directory structure in order to hierarchically group calculated and restricted key figures of an InfoProvider.

Related Information

Creating a Where-Used List for SAP BW/4HANA Objects [page 27]
Using Reusable Query Components [page 379]

2.1.3.7 More Views

You can find more views under *Window* > *Show View* > *Other* > *Business Warehouse*.

Related Information

BW Job View [page 17]

2.1.3.7.1 BW Job View

The *BW job view* in the BW Modeling Tools provides an overview of BW jobs that run in the systems of your BW projects. If the job type supports this, you can perform activities from the view, such as interrupting a job that is running or restarting a failed job.

In the *BW job view*, the following BW job types are supported:

- DATAFLOWCOPY
- REMODELING
- DTP_LOAD (DTP execution using the BW Modeling Tools)
- DTO

By choosing  (*Filter*) in the toolbar of the view, you can specify which jobs are displayed in the view.

In accordance with the selected filters, the jobs are displayed on the left of the view in a list with information on the job type, project (the system in which the job is running), user, start time, status and progress. Any messages about the job are displayed too. If the Job type supports this, you can interrupt a job that is running or restart a failed job. In these cases, a column in the overview of the jobs contains the relevant pushbuttons *interrupt* and *restart* .

By choosing  (*Refresh*), you can refresh the overview in the view.

If you select a job, the details for this job are displayed to the right of the view, together with the job log if one exists.

If the job type supports this, the *BW job view* opens directly when you execute a job directly in the BW Modeling tools. You can also open the *BW job view* by choosing *Window* > *Show View* > *Other* > *Business Warehouse* > *BW Job view* .

Define Filters for the BW Job View

1. Choose  (*Filters*) to define which jobs are displayed in the *BW job view*:
   - *Time Interval*: You can specify which time interval the jobs have to be in.
   - *Destinations*: You can specify which BW projects jobs are displayed for.
   - *Job Status*: You can specify which job statuses are included in the display.
   - *Types*: You can specify which job types are included in the display.
2. Choose *Apply and Close*.

Related Information

Subscribing to the BW Job Repository Feed [page 22]

2.2 Basic Tasks

The following sections introduce the basic tasks for working with the BW Modeling tools.

2.2.1 Working with BW Projects

BW projects form the starting point for working with the BW Modeling tools in the Eclipse-based IDE. In the following sections, you will find detailed descriptions of the basic activities involved in BW projects.

You work with projects in the *Project Explorer* view. The projects are displayed here in alphabetical order. To work in a BW project and see the subtrees of the BW project, you have to log on to the back end system. There are several ways of opening the logon screen:

- By double-clicking the BW project
- By expanding the first level of the BW project

If you double-click or expand a BW project with SSO enabled for the connected BW system, the connection is established without displaying the logon screen.

Once you are logged on to the system for a particular project, you remain logged on for this project until you exit the Eclipse-based IDE or close the project.

2.2.1.1 Creating BW Projects

Before you start working with modeling in the Eclipse-based IDE, you first need to create a BW project. The BW project is used to manage the connection to the back-end system that you want to work with. The project acts as a container (on the front end) for the SAP BW/4HANA metadata objects located in the system.

Prerequisites

- Make sure that you have defined the system that you want to connect to in your local SAP Logon.
- For security reasons, check if the **Secure Network Communication (SNC)** is enabled for the selected system connection. If SNC is not enabled, use the SAP Logon Pad to enable the option.

Procedure

1. To launch the project creation wizard, choose *File* > *New* > *Project* in the main menu.
2. Select the project type *Business Warehouse* > *BW Project* and choose *Next*.

3. In the *Connection* field, enter the system ID or choose *Browse* to select a system from SAP Logon. Then choose *Next*.

   > → Tip
   >
   > When entering the system ID, you can access the content assist function by pressing `Ctrl + Space`.

4. Enter the logon data, for example *Client*, *User* and *Password* (not required for Single Sign-On).
5. [Optional:] If you want to change the default language, select the *Change Default Language* checkbox and select a language.

   > ⓘ Note
   >
   > The default language predefines the original language of the new SAP BW/4HANA metadata objects that will be created in the corresponding system. Consequently, all new metadata objects will automatically be created in the default language. Your BW project might contain objects in different original languages however if you copy objects from other systems for example.

   > → Tip
   >
   > You can access the content assist function by pressing `Ctrl + Space`.

6. [Optional:] If you want to define a project name, choose *Next* and enter a project name. If you do not enter a project name, the project name is generated when the BW project is created.
7. [Optional:] If you want to connect a SAP HANA database to your BW project, choose *Next* and select the SAP HANA system.
8. Choose *Finish* to create the BW project. If you have not entered a project name, a name is generated using the following naming convention: <system ID>_<client>_<user>_<language>_<identifier for duplicate names>. Example: ABC_000_MUELLER_DE or ABC_000_MUELLER_DE_2.

Results

To check your created project, switch to the Project Explorer and expand the first level of the project structure. Verify that the project structure includes the following nodes:

- Favorites
- BW Repository
- Data Sources
- SAP HANA System Library - (if you have attached a SAP HANA system to your project)

2.2.1.2 Attaching SAP HANA to a BW Project

You can use SAP HANA views of the SAP HANA database, on which the BW system is running, in SAP BW/4HANA metadata objects. In order to enable the consumption of SAP HANA views (analytic or calculation

views) in SAP BW/4HANA metadata objects, you have to attach the corresponding SAP HANA system to the BW project.

Prerequisites

- The SAP HANA database has been added as a system to the SAP HANA system view in the SAP HANA administration console.
- Make sure that a database user has been assigned to the SAP HANA system.

Procedure

If you want to attach a SAP HANA database to a new BW project, proceed as follows:

1. In the BW Project creation wizard on the *SAP HANA System* page, select the *Attach SAP HANA System* checkbox.
2. Select the system and choose *Finish*.

If you want to attach a SAP HANA database to an existing BW project, proceed as follows:

3. Double-click the BW project to log on to the back end system.
4. Choose *Attach SAP HANA System* in the BW project's context menu.
5. Select the SAP HANA system and choose *Finish*.

Results

The attached SAP HANA system is displayed as the last child node of the BW project in the Project Explorer tree.

> → Tip
>
> You can remove the system again by choosing *Remove SAP HANA System* in the BW project's context menu. The SAP HANA system is then no longer displayed in the BW project tree.

2.2.1.3 Closing BW Projects

If you do not want to process a project as an active project, you can close it. When a project is closed, its sub-nodes no longer appear in the *Project Explorer* view, but they still reside in the local file system. Closed projects require less memory.

Procedure

1. Select one or more BW projects.
2. Open the context menu and choose *Close Project*.

Results

The project node is still visible in the *Project Explorer* view. This means that you can reopen the project whenever you need it.

→ Tip

- You can close an open connection to BW system (log off) by closing a BW project.
- You can reopen a BW project by selecting it in the *Project Explorer* view and choosing *Open Project* in the context menu.
- You can hide closed projects in the *Project Explorer* menu by using the *View Menu* > *Customize View* function.

2.2.1.4 Deleting BW Projects

If you no longer need a BW project, you can delete it. In the *Project Explorer* view, you can also delete projects of other project types (such as ABAP projects).

Procedure

1. Select one or more projects.
2. From the context menu, choose *Delete*.
3. A dialog box opens. If you want to delete the project contents on disk, select the relevant checkbox. Note that this deletion cannot be undone.
4. [Optional:] Choose *Preview* to check your project selection again and make any changes required. On the following page, you can deselect projects. When you press *OK*, the selected projects are deleted.

5. Choose *OK*.

Results

Once the BW Project has been deleted, it is removed from the *Project Explorer* view.

2.2.1.5 Subscribing to the BW Job Repository Feed

If you subscribe to the *BW Job Repository* for a SAP BW/4HANAsystem, jobs of certain SAP BW/4HANA job types are displayed in accordance with the specified filter criteria in the *Feed Reader*.

Context

For the *BW Job Repository* feed, the following job types are supported:

- DATAFLOWCOPY
- REMODELING
- DTP_LOAD (DTP execution using the BW Modeling Tools)
- DTO

For general information about the *Feed Reader* view and about subscribing to feeds in the *Feed Reader* view, see *ABAP Development User Guide*.

Procedure

1. Open the *Feed Reader* view. If it is not visible yet in the BW modeling tools, open it by choosing *Window* > *Show View* > *Other* > *ABAP* > *Feed Reader*.
2. In the toolbar for the view, choose  (*Add Feed from Repository*).

   The *Feed Query* wizard is displayed.
3. To keep feed entries for a particular SAP BW/4HANA system, select the relevant BW project on the left in the first wizard step and then select the *BW Job Repository* feed on the right.
4. Choose *Next*.
5. In the next wizard step, configure the feed query for the *BW Job Repository* feed.
   a. If necessary, change the title of the feed.
   b. If you want to receive popup notifications whenever there are new entries for the feed, select *Display Notification Messages*.
   c. By choosing *Refresh Interval*, you can specify how often the client should search for new entries. .

d. By choosing *Paging*, you can specify the number of feed entries after which there should be a page break in the *Feed Reader*.
e. Set a filter to restrict the jobs to be displayed.
    1. In the default setting, the system displays all jobs scheduled by the current user and with an age of up to and including 86,000 seconds (1 day). You can define your own rules for filtering however. For example, you can filter according to user executing a job, job status and job type.
    2. You can specify that a feed entry is created if all defined filter rules are complied with, or that an entry is created if just one of the rules is complied with.

6. Choose *Finish*.

Results

The *BW Job Repository* feed is displayed in the *Feed Reader* view below the project selected in the feed query.

By choosing  (*Refresh selected feeds*), you can refresh the feed(s) selected in the *Feed Reader* view.

Next Steps

In the *BW job view* you can perform activities such as pausing a job that is running or restarting a failed job (provided that the job type supports this).

Related Information

BW Job View [page 17]

2.2.2 Working with BW Metadata Objects

In the *BW Modeling* perspective, you can open and edit all SAP BW/4HANA metadata objects that are displayed in the BW projects. In the case of object types for which there is no native Eclipse editor, the SAP GUI editor is opened in Eclipse-based IDE in an editor view. Process chains are opened in the *Process Chain Editor* app once the SAP BW/4HANA Cockpit has been configured.

2.2.2.1 Opening SAP BW/4HANA Metadata Objects

There are various ways of opening metadata objects:

- By double-clicking the object in the *Project Explorer* view.

- By opening the context menu for one or more objects in the Project Explorer tree and choosing *Open*.
- By using the *Open BW Object* dialog box.

> ⓘ Note
>
> When a user opens a native Eclipse editor for a metadata object, the object is not locked. The object will only be locked once the user starts working with it.

2.2.2.2 Searching for and Opening Objects with the Open BW Object Dialog Box

Using the *Open BW Object* dialog box, you can search for a SAP BW/4HANA metadata object by name and by description, and then open the editor of the selected object. Process chains open in the *Process Chain Editor* app once the SAP BW/4HANA Cockpit has been configured.

Procedure

1. In the *Project Explorer* view, select the required BW project.
2. Log on to the connected system (not required for SSO).
3. In the main menu, choose *Navigate* > *Open BW Object* or the *Open BW Object* pushbutton in the toolbar of the *BW Modeling* perspective.

   > ⓘ Note
   >
   > If you have not selected a project before, the *Open BW Object* dialog box will suggest the first BW project listed in Project Explorer.

   > → Tip
   >
   > In the *Open BW Object* dialog box, you can also select another project or create a new one.
4. Enter a search string.

   The system searches for metadata objects in the selected BW project. It searches for objects where the search term occurs in or matches the prefix of the technical name or the description.

   The search is not case sensitive. You can run the following wildcard searches:

   - ? - any character
   - * - any string (including the string length 0)

   The search results are listed in the dialog box under *Matching Items*. They are sorted according to object type and description in ascending order.

   > ⓘ Note
   >
   > - There is a maximum number of matching items in the dialog box. The default value is 200. You can customize this value in the BW modeling tools settings.

- Under *Matching Items*, you can only see metadata objects that you have at least display authorization for.

5. Alternatively you can first select an object type and then enter the search term.
   a. To do this, enter **`type`** under *Select an item to open* and confirm your entry.
   b. Select the required object type from the list by double-clicking it.
   c. Enter the search term.
6. In the search results, select the object that you want to open.
7. Choose *OK*.

Results

The editor of the selected SAP BW/4HANA metadata objects opens. Process chains open in the SAP BW/4HANA Cockpit in the Web Browser once the SAP BW/4HANA Cockpit has been configured.

2.2.2.3 Searching for SAP BW/4HANA Objects

The BW search allows you to search for SAP BW/4HANA objects using various search criteria - either in a specified BW project or system-wide. The function returns a hit list in the *Search* view. You can also use the BW search to compile a where-used list for SAP HANA objects in SAP BW/4HANA objects.

Procedure

1. Choose *Search* > *Search...* in the user menu. As an alternative, you can use the shortcut `Ctrl + H` or press the *Search* () button in the *BW Modeling* perspective.
2. In the popup that now opens, go to the *BW Object Search* tab page.

   → Tip

   If you cannot see this tab page, press the *Customize...* button and select it to be displayed.
3. Enter your search term in the *BW Object Search* tab page.

   The system searches for objects where the search term occurs as a prefix of the technical name or the description (see next step), or where the search term matches the technical name or the description (see next step).

   The search is not case sensitive. You can run the following wildcard searches:

   - ? - any character
   - * - any string (including the string length 0)
4. Unter *Name* and *Description*, specify whether to search by technical name, description or both.

5. Under *BW Project*, select an active project that you want to search in.
6. Define further restrictions for the search.

Option	Description
Active Objects / Inactive Objects / Content Objects	You can define whether to search for active objects and/or inactive objects. content objects (D version).
Object Type	You can restrict the search to one or more object types. Restriction of the search to subtypes is only possible for certain object types, such as InfoObjects. Under *Object Type* or *Subtype*, enter a comma and select the required object type from the list by double-clicking it. To restrict by multiple object types, use commas as separators.
Created From/To	You can restrict the search to objects created during a specified time period.
Created By	You can restrict the search to objects created by a specified user.
Last Change From/To	You can restrict the search to objects changed during a specified time period.
Last Change By	You can restrict the search to objects last changed by a specified user.
Uses Object	If you choose *Browse*, the *Uses HANA Object* dialog box appears. Here you can select an SAP HANA object by using a search string. The result of the BW object search is restricted to those SAP BW/4HANA objects, which use the selected SAP HANA object. This enables you to use the BW modeling tools to compile a where-used list for SAP HANA objects in SAP BW/4HANA objects. For example, you can check which CompositeProviders a specific SAP HANA view is used in.

7. To start the search, choose *Search*.

Results

All objects that match your search criteria are displayed in the *Search* view. You can navigate to the maintenance screen for an object by double-clicking on it in the list, or call various other functions from the context menu.

2.2.2.4 Creating a Where-Used List for SAP BW/4HANA Objects

The where-used list allows you to find all objects that use a certain object. The function returns a hit list grouped by SAP BW/4HANA object type.

Procedure

1. You can start the where-used list from the following places:
   - For all objects that are displayed in the BW modeling tools: Project Explorer
   - For objects that are maintained using a native Eclipse editor: Active Editor
   - For InfoObjects and reusable query components: *InfoProvider* view
2. You can call the where-used list as follows:
   - For objects that are displayed in the native Eclipse editor: Press the  (*Where-Used*) button in the *BW Modeling* perspective toolbar.

     > **Note**
     >
     > The pushbutton is not active for objects that the SAP GUI editor opens for in the Eclipse-based IDE, and which do not have a native Eclipse editor. Open the where-used list for these objects in Project Explorer.

   - For objects in the Project Explorer and *InfoProvider* view: Open the context menu for the object and choose *Where-Used*.

Results

The list of usages is displayed in the *Search* view. The list of usages displays the hits grouped by object types. By double-clicking on an object in the hit list, you can navigate to the object's maintenance screen.

2.2.2.5 Opening a SAP BW/4HANA Object in another Project

With this function, you can navigate rapidly between two versions of the same SAP BW/4HANA object in two different projects.

Procedure

1. In the context menu of the object in question in *Project Explorer*, choose *Open in Project*.
2. Select the BW project that you want to open the object in.

   The system shows you all open BW projects apart from the current package. Note here that projects that do not contain the object in question can also be displayed.

   > ⓘ Note
   >
   > If the project that you want to open the object in does not exist yet, you can create it by choosing *New Project*.

Results

In the Editor, the object is opened in the system of the selected BW project. If the object is not contained in the selected BW project, the system displays an error message.

2.2.2.6 Creating a Copy of a SAP BW/4HANA Object

For SAP BW/4HANA objects with native Eclipse editors, you can create a copy.

This can be done using the following functions:

- For saved objects:
  - In the Project Explorer using the *Copy* option in the context menu of the object.
  - In the wizard for creating an object with the *Copy From* function.
- From the editor by choosing *File* > *Save As...* .

  > ⓘ Note
  >
  > To make sure that the texts are copied in all languages when saving InfoObjects under a different name, install SAP Note 2326441.

For data flow objects, enhanced copy options are provided. You can create multiple target data flows and specify changes for the target data flows in comparison to the copied data flow.

Related Information

Copying a Data Flow [page 210]

2.2.2.7 Managing Favorites

For a better overview of the objects that you work with, you can set them as favorites.

Related Information

Adding Objects to Favorites Using the BW Repository Tree [page 29]
Adding Objects to Favorites Using the Favorites Tree [page 30]
Removing Objects from the Favorites [page 31]
Removing Objects from the Favorites [page 31]

2.2.2.7.1 Adding Objects to Favorites Using the BW Repository Tree

You can save objects from the *BW Repository* tree in favorites. Favorites are saved as cross-project favorites in the back end system.

Procedure

1. Double-click the BW project to log on to the back end system.
2. Select one or more InfoAreas or SAP BW/4HANA metadata objects in the *BW Repository* tree.
3. From the context menu, choose *Add to Favorites*.

Results

If you add complete InfoAreas to the favorites, they are added as children of the favorites node.

If you add metadata data objects to the favorites, they are added as children of the semantic folders that correspond to the relevant object type (for example, *CompositeProvider* or *DataStore Object (advanced)*). If the semantic folder does not exist, it is created as a child of the favorites node.

→ Tip

Choose *Refresh* from the context menu in order to read current information from the back end system and display it in the BW project.

2.2.2.7.2 Adding Objects to Favorites Using the Favorites Tree

You can use the *Favorites* tree to add objects to the Favorites. Favorites are saved as cross-project favorites in the back end system.

Procedure

1. Double-click the BW project to log on to the back end system.
2. In the context menu of the *Favorites* tree, choose *Add Objects*.

   The *Add Objects to Favorites* screen appears.
3. Enter a search term.

   The system searches for SAP BW/4HANA metadata objects in the BW project. It searches for objects where the search term occurs in or matches the prefix of the technical name or the description.

   The search is not case sensitive. You can run the following wildcard searches:

   - ? - any character
   - * - any string (including the string length 0)

   The search results are listed in the dialog box under *Matching Items*. They are sorted according to object type and description in ascending order.

   ⓘ Note

   - There is a maximum number of matching items in the dialog box. The default value for this is 200. You can change this value in the settings for the BW Modeling tools.
   - Under *Matching Items*, you can only see metadata objects that you have at least display authorization for.
4. Select one or more objects and choose *OK*.

Results

If you add complete InfoAreas to the favorites, they are added as children of the favorites node.

If you add metadata data objects to the favorites, they are added as children of the semantic folders that correspond to the relevant object type (for example, *CompositeProvider* or *DataStore Object (advanced)*). If the semantic folder does not exist, it is created as a child of the favorites node.

> → Tip
>
> Choose *Refresh* from the context menu in order to read current information from the back end system and display it in the BW project.

2.2.2.7.3 Removing Objects from the Favorites

Procedure

1. Double-click the BW project to log on to the back end system.
2. Select one or more InfoAreas or metadata objects in the *Favorites* tree.
3. Open the context menu and choose *Remove from Favorites*.

Results

When you remove objects from the favorites, they are no longer visible in the favorites for other BW projects connected to the same system.

2.2.2.7.4 Removing Objects from the Favorites

Procedure

1. Double-click the BW project to log on to the back end system.
2. In the context menu of the *Favorites* tree, choose *Remove All*.

Results

The system removes all objects from the favorites, without a security prompt. The objects are then no longer visible in the favorites for other BW projects connected to the same backend system.

2.2.2.8 Linking a Selection in the Project Explorer View with the Active Editor

In the *Project Explorer* view, you can precisely locate the SAP BW/4HANA metadata object, which you are currently working on, in the *BW Repository*.

Procedure

In the Project Explorer toolbar, press  (*Link with Editor*).

Results

The selection in the *BW Repository* tree of the *Project Explorer* view is linked to the active editor. If you switch to the active editor, the metadata object currently being edited is automatically selected in the *BW Repository* tree.

2.2.2.9 Sharing Links to SAP BW/4HANA Objects

You can share a given SAP BW/4HANA object by e-mail or using a document of your choosing. To do this, you can create a hyperlink to this object in the BW Modeling tools. The recipient can use this to open the referenced object in another installation of the BW Modelign tools in the corresponding BW project.

Prerequisites

- The Eclipse-based IDE with the BW Modeling tools must be open when the link is opened.
- You have access authorization for the corresponding backend system and authorizations for the linked object.

Procedure

1. Select the object in the *Project Explorer* and choose *Share BWMT Link* in the object's context menu, or choose (*Share BWMT Link*) in the object's native Eclipse editor.
2. In the *Share BWMT Link* dialog box, select one of the following options:
   - To create an e-mail using your default e-mail application: *E-Mail Link*
     An e-mail is opened, containing a generated subject and the hyperlink to the object as text.

- To copy the hyperlink to the clipboard: *Copy link to clipboard*
  Insert the link into a document of your choosing. The hyperlink is inserted into the document as text.

Results

The following example shows the link for a given CompositeProvider: bwmt://<BW-System-ID>/sap/bw/modeling/hcpr/ztest1/m

You can also create a system-independent link by pressing the shift key in the *Share BWMT Link* dialog box *E-Mail Link* and choosing *E-Mail Link* or *Copy Link to Clipboard* wählen. The link for a specific CompositeProvider then looks like this: bwmt://Systemid/sap/bw/modeling/hcpr/ztest1/m

> **Note**
>
> For the hyperlink to react to the right mouse click, the recipient must specify which application to start when a hyperlink is opened.

For a BWMT link from an external document or an e-mail to be opened, an installation of the BW Modeling tools must be defined as the installation where the object in question should be opened. Under *General* > *Link Handlers*, you can set the flag for the *bwmt* schema (BW Modeling Tools).

Alternatively, you can insert the link into the clipbard and choose *Navigate* > *Open BWMT Link From Clipboard* in the Eclipse menu. This function then opens the object referenced in the link directly.

If you open a system-independent link, the system shows you the BW projects that are open in the installation of the BW Modeling tools, and you can select the relevant project. Note here that projects that do not contain the object in question can also be displayed.

2.2.2.10 Reset SAP BW/4HANA Object to Active Version

You can reset a modified and saved object to the most recently activated version and thus delete the currently revised version. This function is available for InfoObjects, DataStore Objects and CompositeProviders.

Context

In the Eclipse Editor toolbar, the version of the object currently displayed can be seen to the left of the  (*Show in History View*) button. This function is available for objects in revised version (M version), for which an active version exists.

Procedure

Choose the *Reset to Active Version* function by clicking the arrow to the right of the  (*Show in History View*) button.

2.2.2.11 Comparing Object Versions

You can compare two versions with one another. You can compare two versions of an object in the same system or a version of an object in one system and the active version of the same object in a different system.

The version comparison is currently supported for CompositeProviders, DataStore objects (advanced) and InfoObjects.

Compare two versions of an object

1. Press the  *Show in history view* pushbutton in the object editor toolbar to call the *History* view for the object.
2. Select two versions and choose *Compare* from the context menu.
3. In the *BW Object Compare* view in the upper part of the screen, the system displays all differences in the newer version.
4. Click on an entry to display the difference between the two versions in detail in the lower part of the screen.

Compare the version of an object with the A version of an object from a different system

If you want to compare similar objects in different systems, you can compare the active or historical version in the active editor with an active version in a different system.

1. Open the required version of the object. For the version comparison, the object or the version in the active editor is used.
2. In the toolbar in the BW Modeling tools, press  *Compare*.
3. In the *Select Compare Target* dialog box under *BW Project*, select the system where the object that you want to compare the version in the active with is located.
4. Select the object for the comparison.
5. Choose *OK*.
6. In the *BW Object Compare* view, in the upper part of the screen, the system displays all differences in the first object of the version comparison.
7. Click on an entry to display the difference between the two versions in detail in the lower part of the screen.

2.3 Reference

Related Information

Keyboard Shortcuts for BW Modeling Tools [page 35]
Display Options for the BW Modeling Tools [page 36]
Settings for BW Modeling Tools [page 37]

2.3.1 Keyboard Shortcuts for BW Modeling Tools

For an overview of the keyboard shortcuts that you can use in BW modeling tools, choose *Help* > *Show Active Keybindings*.

The following keyboard shortcuts are specific to BW modeling tools:

Open BW object	Ctrl + Shift + D
Activate BW object	Ctrl + F3
Check BW object	Ctrl + F2
Search for BW object	Ctrl + H
Select a BW wizard	Ctrl + N

More helpful keyboard shortcuts for working in Eclipse:

Refresh active editor	F5
Save object	Ctrl + S
Undo last change	Ctrl + Z
Apply last change again	Ctrl + Y
Create new object	Ctrl + N
Close Editor	Ctrl + W
Switch between open editors	Alt + Left-Arrow

2.3.2 Display Options for the BW Modeling Tools

You can use various options to display the content of BW modeling tools just how you want.

Link with Editor

The *Link with Editor* button can be found in the toolbar in the *Project Explorer* view. It toggles the link on and off between the *Project Explorer* view selection and the active editor.

Switch Label Display Mode

The *Switch Label Display Mode* button in the toolbar in the *BW Modeling* perspective allows you to switch between various options, to display the technical names (key) and/or descriptions (text) of InfoAreas, SAP BW/4HANA metadata objects and elements of the CompositeProvider. The default setting is <[key]text>.

> **Note**
>
> Note that this setting is not used in the *Open BW Object* dialog.

Sort Project Explorer Tree

You can alphabetically sort the folders and objects in a BW project in the *Project-Explorer* view. To do this, choose *Alphabetical Order for BW Objects* in the project's context menu.

Once you have selected the entry, the folders in the trees in the *Project Explorer* view are displayed in alphabetical order. Sorting is applied in accordance with the *Switch Label Display Mode* setting on the technical names or the description of the folders.

SAP BW/4HANA metadata objects are displayed in alphabetical order by technical name.

2.3.3 Settings for BW Modeling Tools

Use the dialog box under *Window* > *Preferences* to specify how you want Eclipse to work. Here you can configure a number of settings for BW Modeling tools.

BW Modeling

You can change the following settings on the *BW Modeling* preferences page:

- **Search and Validation Settings**:

Option	Explanation	Default Value
Max. Number of Entries in Result Set	Specifies how many entries are displayed as matches (after a defined latency) in the Open BW Object dialog box and in the wizards for creating SAP BW/4HANA metadata objects.	200
Input delay (msec)	Specifies the latency time that elapses before entries are displayed as matches in the Open BW Object dialog box and in the wizards for creating SAP BW/4HANA metadata objects.	800

- **Result Set Sorting**:
  By default, the result lists in the BW Modeling Tool are sorted firstly by object type, then by name and then by description. You can use the *Up* and *Down* buttons to change the order of the criteria for sorting. If you are looking for a string in the name for example, and the string in question occurs frequently, it can help to sort by name first, and then by type.
- **Transport Settings**:
  Here you can define the default setting for transport packages. You can either set a global default package or a package specific to a system.
- **Data Protection**:
  If you set the flag for *Maintenance of Anonymization Rules*, you allow anonymization rules to be defined. The system adds an *Anonymization Rules* tab to the editor for editing a suitable InfoProvider. Based on these anonymization rules, you can use the Data Protection Workbench to anonymize personal data.

> ⓘ Note
>
> For more information, see .

Automatic Checks

On the *BW Modeling* > *Automatic Checks* page, you can define which objects the checks should be directly displayed for while working on problems in the Editor in the *Problems* view. You can switch the automatic check

on and off for all objects (*Mode*: *All* or *None*) or explicitly decide which objects should be checked (*Mode*: *Individual*).

CompositeProvider

You can change the following settings on the *BW Modeling* > *CompositeProvider* page.:

Option	Explanation	Default Value
Min. Number of PartProviders for Bulk Load	For more information, see *Settings for CompositeProviders*	10

Formula Syntax Coloring

On the *BW Modeling* > *Syntax Coloring* page, you can define the syntax highlighting for formulas in queries and can set the required colors, styles and fonts.

For more information, see *Defining Formulas*.

Query Designer

On the *BW Modeling* > *Query Designer* page, you can define whether hierarchies are loaded by node or completely up to the specified level. To improve performance, we recommend selecting the *Dynamic Node Loading* field.

Related Information

Sharing Links to SAP BW/4HANA Objects [page 32]
Settings for CompositeProviders [page 311]
Defining a Formula [page 344]

3 Concepts

The following sections explain the concepts brough to you by the BW modeling tools.

3.1 Modeling with Fields Instead of InfoObjects

Objects in SAP BW/4HANA can be modeled with InfoObjects or with fields. Modeling InfoObjects brings additional effort, but offers a number of advantages. When deciding whether to model with fields or with InfoObjects, you should consider how much of the flexibility that the fields offer you you need, and which functions or performance that the InfoObjects offer you you can do without.

You should model central master data objects (navigation attributes and hierarchies) using persistent InfoObjects. With many OLAP functions, this offers you advantages in terms of performance.

In the entry layer in the data warehouse however, where there is little need for OLAP functions, and smaller data volumes are processed, using fields can enhance the flexibility and range of the data warehouse.

Functional restrictions when modeling with fields

- The services for InfoObjects (attributes and hierarchies for example) are not available for fields.
- Fields are displayed in the CompositeProvider / the query with the generated name 4<InfoProvider>-<field name>.
- Validity characteristics for DataStore objects (advanced) with non-cumulative key figures must be InfoObjects.
- InfoObject attributes must be InfoObjects.
- A field-based key figure cannot be an exception aggregation.
- An association with an Open ODS view is only possible once for navigation attributes.
- Planning queries on DataStore objects (advanced) are only supported with fields as read-only.

Performance when modeling with fields

- If the query contains fields, it can be processed key-based in SAP HANA.
- If an open ODS view contains compounded fields, processing in SAP HANA is not possible.
- If fields are used in the query definition, the OLAP cache cannot be used.
- If fields from open ODS view and InfoObjects are used as navigation attributes based on a SAP HANA view in CompositeProviders, performance will be impaired when processing the query filter on them in SAP HANA. More information: 2103032

- In the query on a CompositeProvider, not all data types for fields are supported:
  - The maximum length for fields is 20 characters.
  - Not all characters are supported.

  More information: 2185212
- If fields are used in the query, the InfoProviders can only be read sequentially.

3.2 InfoArea

InfoAreas in the BW Modeling tools serve to structure objects (InfoProviders, InfoObjects, InfoSources and open hub destinations and process chains) in the SAP BW/4HANA metadata repository.

An InfoArea () can contain other InfoAreas as subnodes. In an InfoArea, BW metadata objects are structured according to their object type in semantic folders () such as *CompositeProvider* or *DataStore Object (advanced)*. Semantic folders are only displayed if a metadata object exists for the corresponding object type. InfoArea `0BW4_GLOBAL` - *Global objects* contains all currency translation types, quantity conversion types, key date derivation types, and variables in the system.

The hierarchy of the InfoAreas is displayed in the *BW Repository* tree. In the case of InfoAreas that have been added to the Favorites, the hierarchy beneath them is also displayed in the *Favorites*tree.

> **Note**
>
> As previously, DataSources are structured according to the application components of the corresponding source system.

Related Information

Creating InfoAreas [page 195]

3.2.1 Transporting InfoAreas

The InfoArea is integrated into the SAP BW/4HANA TLOGO framework and can be transported.

The transport object is called AREA.

3.3 Data Flow

A data flow depicts a specific scenario in SAP BW/4HANA. It describes a set of SAP BW/4HANA objects, including their relationships and interdependencies. The BW Modeling Tools contain various editors with graphical user interfaces that enable you to create, edit, document and analyze data flows and their objects.

Explanations of Terms

The following table contains a definition of important terms that you will encounter when working with data flows:

Term	Definition
Data flow	The data flow in SAP BW/4HANA defines which objects are needed at design time and which objects are needed at runtime to transfer data from a source to SAP BW/4HANA and cleanse, consolidate and integrate the data so that it can be used for analysis, reporting and possibly also for planning. A data flow depicts a specific scenario including all involved objects and their relationships.
Data flow object	A data flow object is a TLOGO object in SAP BW/4HANA, which describes a data flow. Data flow objects are created and edited in a graphical editor. They help you to visualize the SAP BW/4HANA objects (and their relationships) contained in a data flow. They have no relevance for the SAP BW/4HANA runtime.
Transient data flow	Using an editor you can flexibly show and analyze the data flow for any persistent, active SAP BW/4HANA object. This representation of a data flow that starts from an object is referred to as a transient data flow. The editor for a transient data flow is referred to as a transient editor. A transient data flow can be saved as a data flow object.
Persistent SAP BW/4HANA object	A persistent SAP BW/4HANA object is an object which has already already been saved in the metadata tables in the database and is independent of the data flow object. A persistent object can be contained in multiple data flows and can therefore be used again in different data flows.
Non-persistent SAP BW/4HANA object	A non-persistent SAP BW/4HANA object is a draft version of a SAP BW/4HANA object. It is an object that only attributes such as object type and name have been specified for so far. It is has not been saved on the database. A non-persistent object can only be displayed and used in the data flow in which it was created. If you create a non-persistent object in the transient editor, it will be discarded if you quit the editor without saving the data flow as a data flow object.

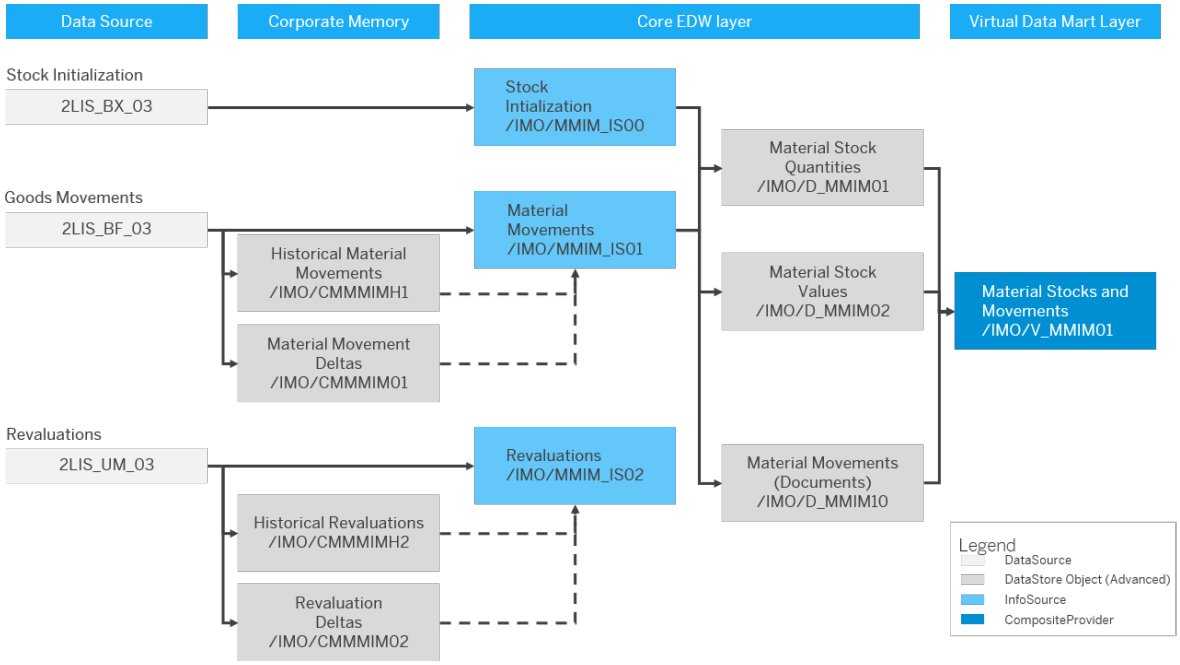
Modeling in the BW Modeling Tools

You can create a data flow object and benefit from its advantages. For example, you can start by creating a logical data flow with non-persistent objects and relationships (blueprint) and then create BW objects, transformations and loading processes later on, in order to persist data in the metadata tables.

You do not necessarily require a data flow object to model objects and relationships in data flows. You can also use the transient editor for this. The graphic transient editor provides a far better overview of a data flow than a tree display can. This means, for example, that when you expand a data flow in different directions, the graphical representation always remains very clear. Even loops, which start from one object via transformations (and sometimes further objects) and return to the initial object, are represented more clearly in the graphical editor than in a tree display.

Examples

The following illustration shows the data flow for the inventory management:



For more examples of data flows, please refer to the BI Content documentation; for example, the section *Supply Chain Management (SAP-HANA-optimized)*.

3.3.1 Data Flow Object

You can model the objects and relationships of a data flow using a data flow object in a graphical editor.

From the **data flow object**, you can create SAP BW/4HANA objects and the connections between them both persistently and non-persistently. This allows you to create a scenario in SAP BW/4HANA from scratch, or

define a template for the technical implementation of a scenario. You can also use the data flow object to store a visualization of an existing scenario with persistent object and existing data flows as a BW object type in SAP BW/4HANA. The data flow object is of no relevance for the runtime.

You do not necessarily require a data flow object to model SAP BW/4HANA objects and relationships in data flows. You can also use the **transient editor** for this.

Advantages of using data flow objects

Graphical modeling of data flow objects offers various advantages, especially in the following use cases:

- Top-down modeling of new data flows
  - Top-down modeling makes quick and structured modeling possible directly in the system.
  - To start with, you can create the logical data flow with all of its elements without having to store the objects on the database. This allows you to create a blueprint of your data models directly in the system. In a subsequent step, you can add the required technical properties to these objects and therefore persist them to the metadata tables in the system.
- Structuring your implementation using data flows
  - You can group modeling objects and store them as a persistent view for your enterprise model. You can also include existing models (from SAP NetWeaver 7.0 or higher).
  - You can get a clearer overview of the data flows by structuring them according to application areas or Data Warehouse layers, for example.
  - You can document your data flows and the objects that they contain.
  - You can use the BW transport connection to collect and transport the objects belonging to a data flow.
  - You can use naming conventions to use and reuse the same models.
- Using data flows as templates to set modeling standards
  - You can model data flows as templates quickly and simply by copying them to your data flow and adapting them.
  - You can use templates to create your own company-wide standards.
  - Using templates can help to reduce development costs.
- Organization of existing models in data flows after upgrading to the current release

Related Information

Creating a Data Flow Object [page 197]

3.3.2 Documenting Data Flows

You can document a data flow object, including the contained persistent and non-persistent objects, in the context of the data flow object (in other words, in the context of the scenario illustrated by the object).

This function allows you to describe the role of a nodes in a specific data flow. In a data flow object, for example, architects can illustrate a business scenario with no-persistent objects, add comments as documentation to nodes and hand over the scenario to technical experts for implementation.

The documentation for the data flow object and its nodes is created and displayed in the *Properties* view. To do this, you can use the text editor and HTML editor provided.

Related Information

Creating Documentation [page 208]

3.3.3 Transient Data Flow

For a persistent object in SAP BW/4HANA, you can display and flexibly analyze the corresponding data flow in the transient editor. In this case, the data flow is referred to as a transient data flow. You can use the transient editor to model objects and their relationships in the transient data flow. You can also save a transient data flow as a data flow object.

The graphical transient editor provides you with a clear display of the data flow for the original object. The graphical display also remains clear if you expand the transient data flow in various directions.

The editor of the transient data flow enables you to find the answers to various questions related to a object and the corresponding data flow. For example:

- Are there currently any outbound transformations and data transfer processes for a specific object?
- Which other objects consume a specific object?
- Where does the data of a specific object come from?

The transient editor can be used to add missing objects in the data flow. Note however that non-persistent objects are discarded when you quit the editor unless you save the data flow as a data flow object.

Related Information

Using a Transient Data Flow for a Persistent Object [page 213]

3.3.4 Transporting Data Flows

The data flow object is integrated into the SAP BW/4HANA TLOGO framework and can be transported. The transport object is DMOD (A version) or DDMO (delivery version).

3.4 Open ODS View

Open ODS views enable you to define data models for objects like database tables. database views or DataSources (for direct access). These data models allow flexible integration without the need to create InfoObjects. This flexible type of data integration makes it possible to consume external data sources in SAP BW/4HANA without staging, combine data sources with SAP BW/4HANA models and physically integrate (load) external data sources by creating DataSources.

The Open ODS view is a SAP BW/4HANA metadata object that provides a structure description with attributes (fields) and data types. It represents a view on a source and adds analytic metadata to this source. Supported data sources are objects, which describe their data in a structure with attributes and data types (such as database tables), views or SAP BW/4HANA DataSources. The Open ODS view does not have separate storage for transaction data or master data. This means persistency and analytic modeling are decoupled for the Open ODS view.

Although classic modeling with InfoObjects guarantees a high level of consistency in the data model, any existing modeling already in use for reporting and analysis is difficult to modify. Modeling with Open ODS views is considerably more flexible. You can create an Open ODS view with minimum effort for a data source and assign properties to the fields of the Open ODS view. In particular, when defining the structure of the Open ODS view, you can specify whether a specific field should be interpreted as a key figure or characteristic. An Open ODS view modeled in this way makes data available for consumption (for example, with a query). The view can be enhanced stage by stage for further integration into BW.

Consuming, Combining and Physically Integrating Data

- Consume external data without staging:
  The Open ODS view allows data sources to be virtually consumed without the need for a separate SAP BW/4HANApersistence for the data. These data models can be located in a database schema (of the SAP HANA database in SAP BW/4HANA), which is not managed by SAP BW/4HANA. Alternatively, the models can be located in other databases, which can be connected to the SAP HANA database in SAP BW/4HANA using SAP HANA Smart Data Access.
- Combine external data with SAP BW/4HANA:
  By using links to InfoObjects, you can combine Open ODS views with SAP BW/4HANA models and thereby use master data attributes and texts in SAP BW/4HANA, for example. There is another way of combining the Open ODS view with SAP BW/4HANA models; by using the Composite Provider.
- Physical integration of external data (staging):
  For ETL purposes, you can create DataSources for Open ODS views and thereby physically integrate data sources into SAP BW/4HANA. In this case, the persistence of the Open ODS view is ensured by a DataStore

object Objekt (advanced), comparable here with a Persistent Staging Area (PSA) in SAP BW. The data can then be processed further in SAP BW/4HANA from within the DataStore object (advanced).

The degree of integration for data models can be iteratively developed, without invalidating any modeling objects (for example, queries) that are based on these modeling objects. For example, you can model a simple Open ODS view with standard properties for the fields. You can then use this view for queries and, in subsequent steps, link InfoObjects, for example, to the fields of the Open ODS view and use the InfoObject properties in your model.

Restrictions

- Open ODS views cannot be defined for hierarchies.

More Information

SAP Note 2198480 provides practical information about using the Open ODS view.

Note the following information: Modeling with Fields Instead of InfoObjects [page 39]

3.4.1 Components of the Open ODS View Definition

An Open ODS view is defined by specifying the following components:

Semantics

- Facts
- Master Data
- Texts

Source Type

The Open ODS view supports various types of data sources (also known as source types):

- SAP BW/4HANA DataSource for direct access
- DataStore Object (Advanced)
- Database table or view
- Virtual table using SAP HANASmart Data Access

- Transformation

The source type is the implementation of the interface of the source object. The source type provides static methods for input help and instantiation.

Source Object

The table or table-shaped object, which the Open ODS view is based on, is known as a source object. A source object instance provides a flat list of fields - with data types and texts - and the access methods for the data. The Open ODS view is a view on the source object. It does not contain any data persistence.

Analytic Metadata (Field Properties)

Analytic metadata is added to the fields of the Open ODS view. For each field, you can specify whether it should be interpreted as a characteristic or as a key figure. Other examples of analytic metadata include the aggregation behavior for key figures, or associations and authorization relevance for characteristics or the InfoObject name of an Open ODS view field. This name is used for the Open ODS view field used across the system, for example in the InfoProvider display of the Open ODS view or in a query on the Open ODS view.

3.4.2 Supported Sources for Open ODS Views

Various sources for data access with Open ODS views are available: DataSources in SAP BW/4HANA, DataStore objects (advanced), database tables or views of the SAP BW/4HANA SAP HANA database, and external sources connected to BW using SAP HANA Smart Data Access. In addition, transformations allow data type conversions, for example, as sources, if data is consumed outside of the SAP BW/4HANA-managed database schema.

Source Types

Database tables or views of the SAP HANA database

With source type *database table or view*, Open ODS views can access data from any schemas on the SAP BW/4HANA SAP HANA database. The connection to the relevant schema is configured in the SAP HANA source system with connection type *Local SAP HANA Database Schema*.

Database tables or views via SAP HANA MDC

SAP HANA supports multiple isolated databases in a single SAP HANA system. These are known as tenant databases. A SAP HANA system that was installed in Multiple Database mode can contain more than one tenant database. For more information about prerequisites and supported database objects, see SAP Note 2312583. - *FAQ:. BW and SAP HANA Tenant Database: Cross-Database Access in SAP HANA Tenant Databases*.

The source type *Database Table or View via HANA MDC* enables Open ODS views to access data in schemas of a tenant database of the SAP HANA system. The connection to the relevant schema is configured in the SAP HANA source system with connection type *SAP HANA Tenant Database Schema*.

Tables and Views from External Sources Using SAP HANA Smart Data Access

With source type *Virtual Tables Using HANA Smart Data Access*, Open ODS views can access remote data from different external sources. The source must be configured in SAP HANA as a remote source. The connection to the source and the relevant schema is configured via a SAP HANA source system with connection type *SAP HANA Smart Data Access* for SAP BW/4HANA.

When an Open ODS view is created with source type *Virtual Table Using SAP HANA Smart Data Access*, a virtual table is created in SAP HANA. The virtual table points to a remote table, which is the source object of the Open ODS view, and allows access to data.

> ⓘ Note
>
> Note that a dedicated source type (Big Data) is available for remote sources with the adapters *Spark SQL DESTINATION* and *VORA (ODBC)*. Therefore remote sources with these adapters are not available for source type *Virtual Tables via SAP HANA Smart Data Access*.

Big Data

The source type *Big Data* enables Open ODS views to access the data of a Hadoop cluster, which is connected via a Big Data source system (and therefore via a remote source with the *Spark SQL DESTINATION* or *VORA (ODBC)* adapters) to the SAP HANA database of SAP BW/4HANA.

DataSources in SAP BW/4HANA

With the source type *DataSource (BW)*, Open ODS views can access the data of DataSources that support direct access. This is possible for ODP and SAP HANA source systems.

Transformations

Using source type *Transformation* you can use transformations for example to perform data type conversions, assignments or string operations on data that is consumed via Open ODS views. This can be useful for example if you cannot or do not want to use these transformations in the data source on data, which is outside of the database schema managed by SAP BW/4HANA.

You can use a transformation as the source for the Open ODS view provided that you have created the corresponding data flow.

DataStore Objects (Advanced)

With the source type *DataStore-Object (Advanced)*, Open ODS views can access data from DataStore objects (advanced). Only DataStore objects for which all reporting takes place on the active table are supported.

Restrictions

Input parameters of SAP HANA views are not visible in the Open ODS view. This applies to both local views and views in the tenant database as well as for access using SAP HANA Smart Data Access. For more information, see SAP Note 2142344.

Related Information

SAP Note 2142344

3.4.3 Suppress Read Access for Unavailable Sources Temporarily

External sources are not always available for read access from an Open ODS view. You can deactivate a source system to suppress access to the source temporarily.

If you use an Open ODS view to directly access a source that currently is not available, this causes an error. This also applies to queries on CompositeProviders that contain an Open ODS view. In the worst case, all queries of a CompositeProvider result in an error if the source for one of the participating Open ODS views is not available. In the source system editor, you can therefore set the affected source system to inactive, thereby suppressing read access to the source temporarily. No data is then displayed for queries on the affected Open ODS views and CompositeProviders. In this case, affected CompositeProviders only display data for other InfoProviders, for example for DataStore objects.

Related Information

Set Source System to Inactive [page 469]

3.4.4 Structure of the Open ODS View Editor

The Open ODS View Editor always contains the *General* tab. Depending on the semantics of the Open ODS view, it can also contain the relevant semantic tabs; *Facts*, *Master Data* and/or *Texts*.

Tab: General

In the *General* tab, you see the name and description of the Open ODS view, together with information about the semantics of the Open ODS view.

Tabs: Facts, Master Data, Texts

The *Facts*, *Master Data* and *Texts* tabs represent the central area for Open ODS view maintenance. Open ODS views for master data can have two types of semantics. In this case, the editor contains both the *Master Data* tab and the *Texts* tab.

The tabs are structured as follows:

Tabs screen area	Content
Upper screen area	Here you see information on the source object. The tooltip can also show further details on the source for certain source types. Certain sources - such as BW DataSource, DataStore object (advanced) and transformation - allow you to navigate to object maintenance.
Left screen area - *Source Fields*	Here you see the structure of the source object including the source fields and their properties. If you are using a transformation as the source, the InfoSource fields are displayed in the structure of the source object.
Central screen area - *View Fields*	Here you see the structure of the Open ODS view with the view fields organized in field categories. The information displayed on the fields includes the following: • The *Associated Object* column shows which association has been configured for the field. • If navigation attributes have been specified for fields, you can expand the list of navigation attributes using the arrow button ▶ on the field name.
Right screen area - *General*	Here you see the analytic properties of the view field for the selected source field/view field.

→ Tip

Enlarge the screen areas *Source Fields* and *View Fields* by dragging the edges of the areas with the mouse. Double-click one of the screen areas to expand it and hide the other screen area. Double-click again to restore the size of the screen area.

Further Information on the Open ODS View

You can display different types of information for the Open ODS view by using the Eclipse views *Properties*, *History* and *InfoProvider*.

The Eclipse view *BW Reporting Preview* opens the default query in the query monitor.

Related Information

Properties View [page 14]
History View [page 15]
InfoProvider View [page 16]
BW Reporting Preview View [page 15]
These views provide further information about your Open ODS view.

3.4.5 InfoObject Names for Fields in Open ODS Views

For every field in an Open ODS view field and the view navigation attributes, InfoObject names are used in the InfoProvider view and in the query. The InfoObject names are derived according to specific rules.

InfoObject Names for Fields in Open ODS Views

Derivation from...	InfoObject name of Open ODS view field
Default The default is active if you have selected *Usage of System-Wide Unique Name*, irrespective of whether or not there is an association for the field of the Open ODS view.	2F<name of Open ODS view>-<name of Open ODS view field> If the Open ODS view is in a namespace: 2F<name of the Open ODS view with generated namespace>-<name of the Open ODS view field>
Association to InfoObject if *Direct Usage of Associated Object by Name* is selected	<name of associated InfoObject>
Association to Open ODS view if *Direct Usage of Associated Object by Name* is selected	2F<name of associated Open ODS view> If the Open ODS view is in a namespace: 2F<name of the Open ODS view with generated namespace>

InfoObject Names for Navigation Attributes of Open ODS View Fields

Direct Usage of Associated Object by Name button	InfoObject name of navigation attribute
selected	<name of associated InfoObject>__<name of navigation attribute> 2F<name of associated Open ODS view>=<name of navigation attribute>
not selected	2F<name of CompositeProvider>-<field name of navigation attribute> 2F<name of CompositeProvider>-<field name of navigation attribute>

3.4.6 Authorization Behaviour for Fields of Open ODS Views

The *Authorization Relevance* property on the characteristic field of the Open ODS view allows you to specify whether or not a field is authorization-relevant. The settings in the Open ODS view also define which analysis authorizations should be checked for the field when the query is executed. When the query is executed, the system also checks whether the user has analysis authorizations.

Which analysis authorizations are checked when the query is executed depends on how the field is set to authorization-relevant.

No Associations Defined for the Characteristic Field: Authorization-Relevance is Defined Locally

If the *Authorization Relevance* field property is set locally, the system checks all authorizations defined for the Open ODS view field.

> **Note**
>
> In the administration screen for analysis authorizations, you can use authorization-relevant Open ODS view fields with their InfoObject name for the authorization definition. The fields are displayed as InfoObjects in the administration screen for analysis authorizations.

Dependency of the Authorization Relevance Property on Associations

If you have defined an association for a field of the Open ODS view, the authorization behavior depends on the settings in the Open ODS view as described below:

Settings in the Open ODS view (Eclipse-Based Editor)	Authorization behavior
You have defined an association and chosen *Direct Usage of Associated Object by Name*.	The authorization relevance is inherited by the association and cannot be changed locally. If the associated object is authorization-relevant, the system checks the analysis authorizations (defined for the associated object) for the Open ODS field view when executing the query.
You have defined an association and chosen *Usage of System-Wide Unique Name*.	The authorization relevance is inherited by the association. If the associated object is authorization-relevant, the system checks the analysis authorizations (defined for the associated object) for the Open ODS field view when executing the query.

Settings in the Open ODS view (Eclipse-Based Editor)	Authorization behavior
You have defined an association, chosen *Usage of System-Wide Unique Name* and used the button  (*Toggle Default Value/Manual Entry*) to overwrite the property (set locally).	The authorization relevance is determined locally. If you have set the field as authorization-relevant, the analysis authorizations defined for the Open ODS view field are checked when query is executed. The Open ODS view field is available together with its InfoObject name as an InfoObject in the administration screen for analysis authorizations. **Note** Even if you do not change the entry after pressing  (*Toggle Default Value/Manual Entry*), the system interprets the property as being specified locally and not as the default or inherited value. This means that the analysis authorizations are checked, which are defined for the InfoObject that represents the field of the Open ODS view in the authorization maintenance screen. The analysis authorizations of the associated object are not checked.

3.4.7 Transporting Open ODS Views

The Open ODS view is integrated into the SAP BW/4HANA TLOGO framework and can be transported. The transport object is FBPA (A version) or FBPD (delivery version). It is important to take certain dependencies into account for the transport.

Dependencies for Source Table or Source View

If you use a table or a view as the source object of an Open ODS view, the table or the view must exist in the target system for the transport, when the open ODS view is transported.

If you use SAP HANA Smart Data Access, the SDA connection must be configured so that the table or view is available in the target system.

Dependencies for SAP HANA Source System

The Open ODS view does not contain any local system information, such as connection information for the database or the database schema. This information is specified in the source system configuration for Open ODS views that access data from database tables or database views via the SAP HANA source system.

In the transport target system, assignment table RSLOGSYSMAP must be correctly configured for SAP HANA source systems. This is used to access tables or views from Open ODS views.

3.4.8 Transformation of Data from External Sources for Open ODS Views

You can use transformations in the SAP BW/4HANA system for various purposes, for example to perform data type conversions or string operations on data, which is consumed using Open ODS views.

This can be useful for example if you cannot or do not want to use these transformations in the data source on data, which is outside of the database schema managed by SAP BW/4HANA. When defining the Open ODS view, you can therefore use source type *Transformation*, with a DataSource as the source and an InfoSource as the target. The InfoSource fields are displayed as source fields for the Open ODS view in the Open ODS view maintenance screen. At runtime, the transformation rules set in the transformation are applied on the fields of the DataSource.

Related Information

Supported Sources for Open ODS Views [page 47]

3.5 InfoObject

Business evaluation objects are known in BW as InfoObjects.

InfoObjects are divided into characteristics, key figures, units, time characteristics and technical characteristics. InfoObjects are the smallest units in BW. They are used to depict information in a structured form required for creating InfoProviders. InfoObjects with attributes or texts can also be InfoProviders in their own right.

Related Information

Creating Characteristics [page 238]
Creating Key Figures [page 265]
Creating a Unit [page 269]
Creating XXL InfoObjects [page 269]

3.5.1 Characteristic

Characteristics are sorting keys, such as company code, product, customer group, fiscal year, period, or region.

They specify classification options for the dataset and are therefore reference objects for the key figures. The characteristics specify the granularity (the degree of detail) at which the key figures are stored in the

InfoProvider. In general, an InfoProvider contains only a sub-quantity of the characteristic values from the master data table. The attributes include the permitted values for a characteristic.

Time characteristics are characteristics such as date, fiscal year, and so on.

Technical characteristics are used for administrative purposes only within BW. An example of a technical characteristic is the request number. This is generated when you load a request as an ID and helps you locate the request at a later date.

Special features of characteristics

If characteristics have attributes, texts or hierarchies, these are known as data-bearing characteristics.

Master data is data that remains unchanged over a long period of time. Master data contains information that is always needed in the same way. References to this master data can be made in all InfoProviders.

A hierarchy is always created for a characteristic. This characteristic is the basic characteristic for the hierarchy (basic characteristics are characteristics that do not reference other characteristics). Like attributes, hierarchies provide a structure for the values of a characteristic. The *company location* is an example of an attribute for *Customer*. You use this to form customer groups for a specific region, for example. You can also define a hierarchy to make the structure of the *Customer* characteristic clearer.

Related Information

Creating Characteristics [page 238]

3.5.1.1 Attribute

There are two types of attributes: Display attributes and navigation attributes.

In the query, display attributes provide additional information about the characteristic. Navigation attributes, on the other hand, are treated like normal characteristics in the query, and can also be evaluated on their own. Attributes are InfoObjects that exist already, and that are assigned logically to a characteristic.

> ✤ Example
>
> Costs of the cost center drilled down by person responsible:
>
> You use the *Cost Center Manager* attribute for the *Cost Center* characteristic. If you want to navigate in the query using the cost center manager, you have to create the *Cost Center Manager* attribute as a navigation attribute, and flag it as a navigation characteristic in the InfoProvider.

3.5.1.2 High Cardinality

The High Cardinality property allows you to create more than two thousand million characteristic attributes for a characteristic.

As a characteristic with high cardinality has a number of restrictions, you should only set this property if you really expect a larger number of characteristic attributes. A characteristic with high cardinality has no persistent SID values and no SID table. This kind of characteristic can therefore only be used in InfoProviders that store the key instead of the SID value of the characteristics it contains.

> ⚠ Caution
>
> Note that you **cannot** use characteristics with high cardinality:
>
> - in SAP HANA analysis processes
> - as a compounding parent
> - as a navigation attribute
> - in hierarchies

3.5.1.3 Master Data

Master data is data that remains unchanged over a long period of time.

Master data contains information that is always needed in the same way. Characteristics can bear master data in the BW system. Master data contains attributes, texts or hierarchies. If characteristics have attributes, texts or hierarchies, these are known as data-bearing characteristics.

3.5.1.3.1 Master Data Types: Attributes, Texts and Hierarchies

Master Data Types

Attributes

Attributes are InfoObjects that are logically subordinate to a characteristic and that cannot be selected in the query.

> Example
>
> You assign the attributes *Person responsible for the cost center* and *Telephone number of the person responsible for the cost center* (characteristics as attributes), as well as *Size of the cost center in square meters* (key figure as attribute) to a *Cost Center*.

XXL Attributes

XXL attributes are XXL InfoObjects, which are logically assigned to the characteristic. You can use XXL InfoObjects to save additional information for a characteristic as data type STRING or XSTRING.

Texts

You can create text descriptions for master data or load text descriptions for master data into BW. Texts are stored in a text table.

> **Example**
>
> In the text table, the *Name of the person responsible for the cost center* is assigned to the master data *Person responsible for the cost center*.

Hierarchies

A hierarchy serves as a context and structure for a characteristic according to individual sort criteria. For more detailed information, see .

Features

Time-dependent attributes:

If the characteristic has at least one time-dependent attribute, a time interval is specified for this attribute. As master data must exist between the period of 01.01.1000 and 12.31.9999 in the database, the gaps are filled automatically.

Time-dependent texts:

If you create time-dependent texts, the system always displays the text for the key date in the query.

Time-dependent texts and attributes:

If texts and attributes are time-dependent, the time intervals do not have to agree.

Language-dependent texts:

You can specify whether texts are language-dependent (in the case of product names: for example, German — Auto, English — car) or not language-dependent (in the case of customer names, for example). The system only displays texts in the selected language.

If texts are language-dependent, you have to load all texts with a language flag.

Only texts exist:

You can also create texts only for a characteristic, without maintaining attributes. When you load texts, the system automatically generates the entries in the SID table.

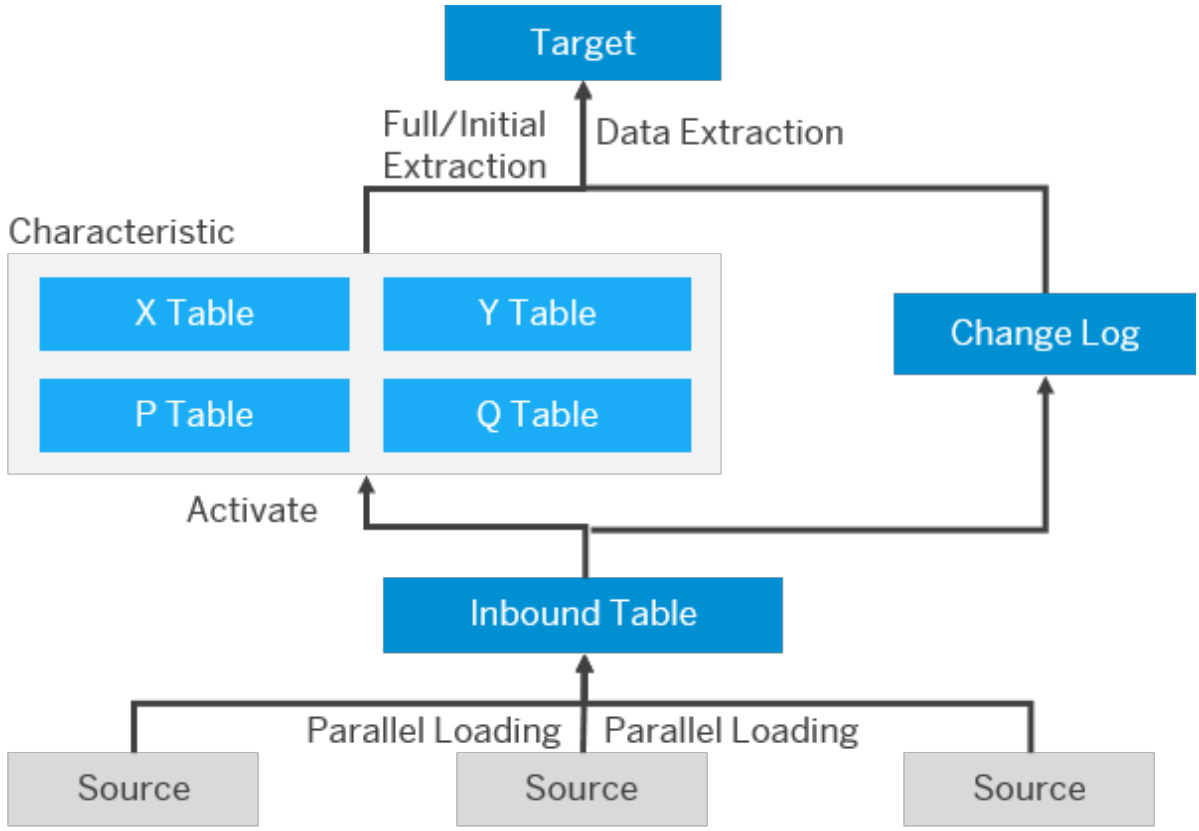
3.5.1.4 Master Data Update

There are two types of master data update. These are the enhanced master data update for loading a large amount of data, and the classic master data update, preferable when only loading a small amount of data.

Enhanced Master Data Update

For characteristics with attributes and/or texts, you can use enhanced master data update. This enables parallel loading of data.

To do this, the attribute receives additional tables. It then has a structure like a DataStore object with an inbound table and a change log. The enhanced master data update is an advantage if you want to load a large quantity of data or to load data from multiple DataSources simultaneously. You can also load deltas from one InfoObject into another InfoObject.



The data is initially loaded in parellel into the inbound table. When activated, the change log and the attribute and text tables of the InfoObject are updated. The activation can be processed in parallel in ABAP runtime, or in the SAP HANA runtime for SAP HANA 2.0 Patch 03 Rev 32.

You can activate enhanced master data update in InfoObject processing. You can also use enhanced master data update for characteristics that have both time-independent and time-dependent attributes (mixed case).

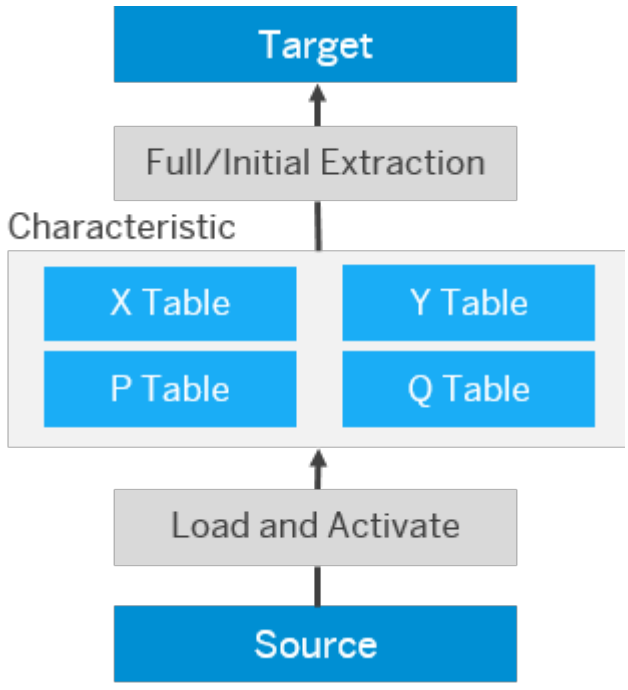
The enhanced master data update is available in the process chain types for activating master data and for removing change log data.

In transaction RSDMD_SETTINGS, you can make settings for processing requests.

> ⚠ Caution
>
> The following restrictions apply:
>
> - In mixed cases, the system always uses ABAP activation.
> - If you use enhanced master data update for characteristics with high cardinality, delta extraction is **not** possible.

Classic Master Data Update

Classic master data update is the default. Note that parallel loading is **not** possible with this solution. If you are only loading only a small amount of data, this solution is preferable. The data is loaded serially into the attribute and text tables directly.



3.5.1.4.1 InfoObject with Write Interface

The data in this InfoObject can be moved to the inbound table using a tool such as Data Services or via SAP Cloud Platform Integration (CPI) by PUSH.

The API for writing to InfoObjects is open for third party tools and can be used by external tools. There is no certification process for this API however.

The familiar master data types support texts and attributes that you use the enhanced master data update for (see Master Data Update [page 58]).

> **Note**
>
> Note that XXL attributes and hierarchies are **not** supported.

Unlike DataStore objects with write interface, InfoObjects with write interface can have up to two inbound tables, one for attributes and one for texts, each with separate requests.

If you have added data activation for the corresponding InfoObject to a process chain, and this process chain was scheduled in streaming, the process chain is triggered when the RSPM request is closed. This means that the entire process chain is processed, including activation of the data. After a short period of time, the data is then available for reporting.

You can use RFC or HTTP data to load data into the InfoObject. The following transfer scenarios are available. You can mix and match these as required.

Scenario	Request Administration	Effort and Complexity
Send Data Without Request (**One Step**)	A new request is opened for each data call.	Easy to implement. Suitable for scenarios with low frequency of data transfer, since otherwise too many requests can be created, thus making housekeeping necessary.
Send Data with Request (**Write into Request**)	The following sequence is used for this: *Open Request*, n times *Send Data* (with request and package), *Close Request*.	Complex implementation, since the sending system (for example the ETL tool) must manage *Open Request*, *Close Request*, and *Send Data* with a consistent ID from the data package. Suitable for scenarios with higher frequency of data transfer.

Scenario	Request Administration	Effort and Complexity
Send Data with Message Request (**Messaging Request**)	A new data package is opened internally for each data call. If no requests are opened, a request is opened when data is sent. A maximum of one request per object is open.	Easy to implement. Suitable for scenarios with higher frequency of data transfer. The request must be closed in the SAP Datasphere, SAP BW bridge system using a process chain. **Note** To do this, you need a process chain in which the *Close Mes. Requests* step (process type `PUSH_MSGR`) is scheduled prior to *inbound activation* so that the request is closed automatically. See . You can use the frequency of the process chain execution to define the number of requests, one request per hour or per day for example. **→ Tip** Note the technical specifications when using the messaging scenario: • For HTTP, pass the **`request=MESSAGING`** parameter when calling the data-Send URI. • For RFC, pass the **`MESSAGING=X`** flag when calling function module `RSO_PUSH_SEND_STRING`.

The data is transferred in the internal format.

Example

A date (data type DATS) is transferred as “20101231”, a time (TIMS) as “235959”.

Fields of type Currency (`CURR`) are transferred in the normal format. During loading, the system moves the decimal place to the standard SAP format.

Example

1234 JPY is transferred -> Due to the currency *Japanese Yen*, the value is saved in internal format as 12.34.

> 12.34 EUR is transferred -> Due to the currency *Euro*, the value is also saved as 12.34.

On fields of type *Character* with conversion routine ALPHA, the system converts to the internal format automatically during loading.

> ### Example
>
> The value “1234” is transferred with conversion routine ALPHA into a field of type *character* with a length of 10 characters. The system automatically converts the value to '0000001234'.

Only after-image deltas can be loaded.

To set this property, go to the corresponding InfoObject in the BW Modeling tools and choose *Properties* > *Enhanced Master Data Update* > *Write Interface-Enabled*.

You can find the URIs specific to each InfoObject to call the functions using HTTP (JSON or XML) in the BW Modeling tools, under *Properties* > *Template URIs*. There are specific URIs for attributes and texts.

The RFC function modules are contained in function group **RSO_PUSH**. These are the same function modules as the ones used for DataStore objects with write interface. When used with the InfoObject, the parameters object_type = IOBJ and object_subtype = ATTR or TEXT must be used.

Related Information

DataStore Object with Write Interface [page 79]

3.5.1.5 Characteristic as InfoProvider

You can indicate an InfoObject of type characteristic as an InfoProvider if it has attributes.

The data is then loaded into the master data tables using the transformation rules. You can define queries for the master data of the characteristic.

Related Information

Editing General Settings [page 239]
Creating Characteristics [page 238]

3.5.1.6 Characteristics of Type Integer

In the SAP HANA database, data often has the format integer (INT4, INT8, DEC). To allow this data to be accessed from the BW system or to be loaded, it must be converted to another format. You can use conversion routine SNUMC to do this.

More information: Conversion Routines in the BW System

You need this conversion for example if you want to use an Open ODS view to access a table or a view in SAP HANA or if you load data into the BW system using a SAP HANA DataSource.

Supported data types

SAP HANA data type	DDIC data type	Internal length	Output length (InfoObject maintenance)	Externe length (SAP HANA DataSource, Open ODS view)	Comment
INTEGER	INT4	11	14	11	
BIGINT	INT8	20	26	20	
DECIMAL(p,0) (p <= 31)	DEC(n,0)	p+1 [2,32]	(4 * (p+1) -2) DIV 3	Same as the internal length	- Fractional units are not supported - Maximum length in the DDIC: 31

3.5.2 Key Figures

The key figures provide the values that are reported on in a query.

Key figures can be quantity, amount, or number of items.

Special Features of Key Figures

A key figure is assigned additional properties that influence the way that data is loaded and how the query is displayed. This includes the assignment of a currency or unit of measure, setting aggregation and exception aggregation, and specifying the number of decimal places in the query.

Related Information

Creating Key Figures [page 265]

3.5.3 Reference InfoObjects

If an InfoObject has a reference InfoObject, it has its technical properties.

For characteristics these are the data type and length as well as the master data (attributes, texts and hierarchies). The characteristic itself also has the operational semantics. For key figures, these are the key figure type, data type and the definition of the currency / unit of measure. However, the referencing key figure can have another aggregation.

These properties can only be maintained with the reference InfoObject.

Several InfoObjects can use the same reference InfoObject. InfoObjects of this type automatically have the same technical properties and master data.

The operational semantics, that is the properties such as description, display, text selection, relevance to authorization, person responsible, constant, and attribute exclusively, are also maintained with characteristics that are based on one reference characteristic.

Related Information

Creating InfoObjects [page 237]

3.5.4 Unit

Units are required so that the values for the key figures have meanings.

Key figures of type *amount* are always assigned a currency key and key figures of type *quantity* also receive a unit of measurement.

Related Information

Creating a Unit [page 269]

3.5.5 XXL InfoObject

You can use XXL InfoObjects to save additional information for a characteristic as data type STRING or XSTRING.

You can further specify the data type by using a MIME type. Numerous formats are supported including different document types, audio files or video files, texts and images.

Related Information

Creating XXL InfoObjects [page 269]

3.5.6 Authorizations for InfoObjects

To work with InfoObjects, you need authorization object S_RS_IOBJA.

3.5.7 Transporting InfoObjects

The InfoObject is integrated into the BW TLOGO framework and can be transported.

The transport object is called IOBJ.

3.6 InfoSource

A structure consisting of InfoObjects without persistence for connecting two transformations.

You always use an InfoSource when you want to perform two (or more) transformations consecutively in the data flow - without additional storage of the data.

If you do not need two consecutive transformations, you can model the data flow without an InfoSource. A transformation is used to write data from the source to the target.

It might be necessary to use one or more InfoSources however due to semantic reasons or the level of complexity. For example, you can specify the format and assignment to InfoObjects in the first transformation. The actual business rules can be performed in the next transformation. If the rules are very complex and interrelated, it might be advisable to use more than one InfoSource.

More information: Recommendations on Using InfoSources [page 67].

InfoSource with Key

You can mark the InfoObjects of the InfoSource as keys. This function is provided for special situations. You should check whether the result really is the one you want.

The way keys are used in InfoSources depends on integration into the corresponding load scenario. If you set the *Aggregation* flag in the definition of the InfoSource, the incoming data will be aggregated according to this key.

This is generally not necessary however, as the transformation in the path with the actual InfoProvider as its target compresses the data by data package anyway in accordance with the keys.

If you flag InfoObjects of the InfoSource as keys, it is very important for this key definition to be compatible with the load scenario that this InfoSource is used in. If the granularity of the key between the InfoSource and the target InfoProvider is not identical, incorrect data might arise. With aggregation SUM, this does not matter. With overwriting however, it is important that overwriting takes place for the correct key combinations.

The InfoSource key must therefore have at least the same granularity as the connected InfoProvider. It is important here to be aware of complex transformations that could change the values in the key fields.

> ### Example
>
> Example of a simple assignment:
>
> If the data target has key characteristics A and B, and characteristics A, B and C can be selected as keys in the corresponding InfoSource, at least characteristics A and B must be selected as keys in the InfoSource.

In complex transformation scenarios, deriving a compatible key definition from the InfoSource might be different, or even impossible.

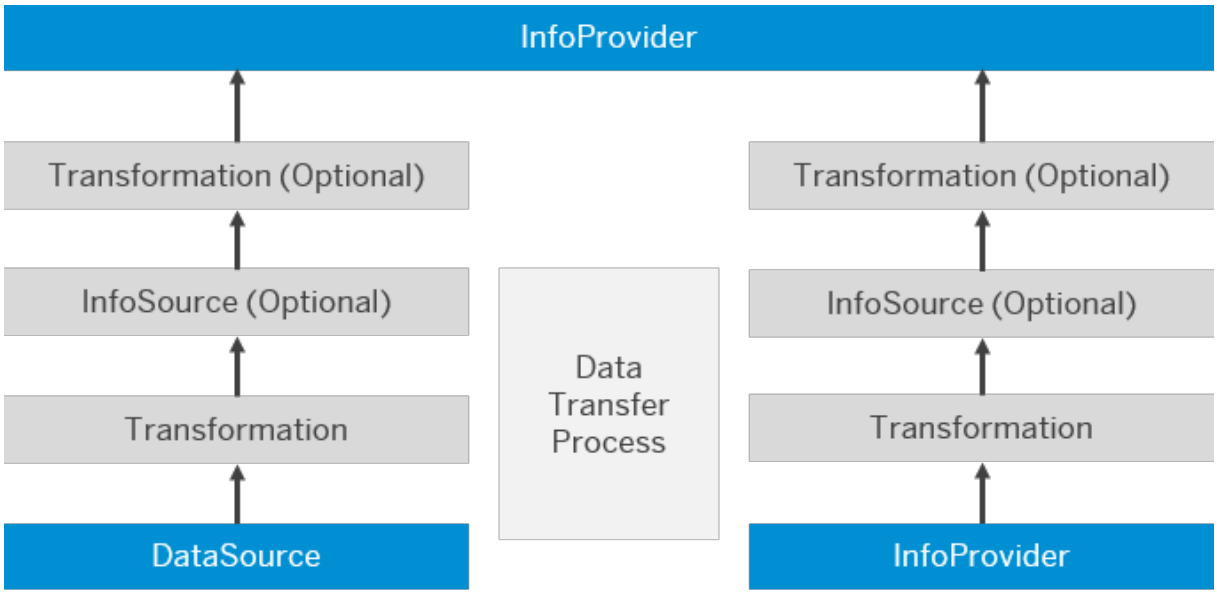
DataStore Object (classic)-Like InfoSource

With a DataStore object as the target of the transformation, you can define the InfoSource as similar to a DataStore object (classic). The InfoObject 0RECORDMODE is then added to the InfoSource. The aggregation behavior is the same as for a DataStore object (classic).

The output from the InfoSource is then the same as if the data were written to the table of new data for a DataStore object (classic). In the case of a reverse image, a record is created with RECORDMODE = R, which results in the record being deleted when the request is activated. If a prior record has already written an after image with the same key in the same data package however, this will be deleted from the data package, and just the reverse image will be passed on to the transformation.

Integration

The following graphic shows how the InfoSource is integrated into the dataflow:



Create the data transfer process from the DataSource to the InfoProvider. Since the InfoSozrce is not a persistent data store, it cannot be used as a target for a data transfer process. Create each transformation on the InfoProvider (as a target) with the InfoSource (as a source) and on the InfoSource (as a target) with the DataSource (as a source).

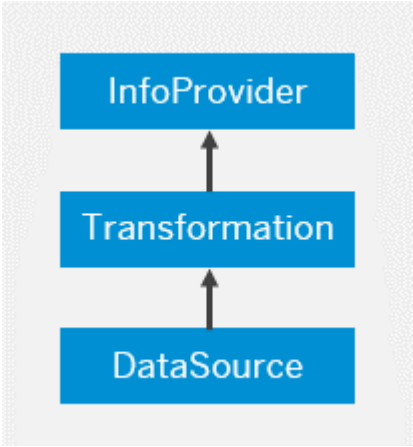
3.6.1 Recommendations for Using InfoSources

Use

This section outlines three scenarios for using InfoSources. The decision to use an InfoSource depends on how the effort involved in maintaining the InfoSource and any potential changes in the scenario can be minimized.

1. Data Flow Without an InfoSource:

The DataSource is connected directly to the target by means of a transformation.



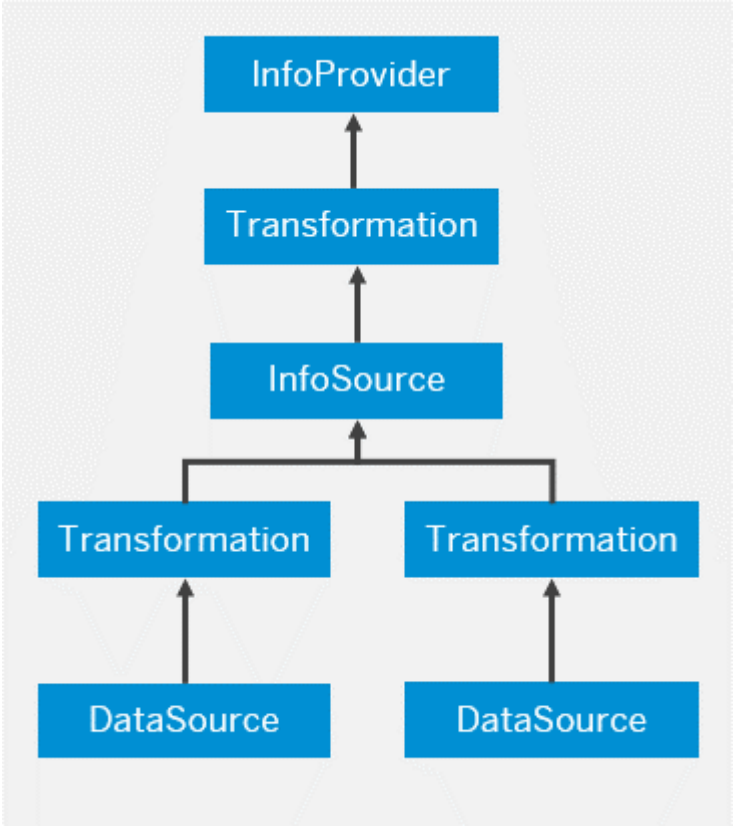
Since there is only one transformation, performance is better.

However, if you want to connect multiple DataSources with the same structure to the target, this can result in additional maintenance effort for the transformation, since you need to create a similar transformation for each DataSource.

You can avoid this if the DataSource is the same, it just appears in different source systems. In this case, you can use source system mapping when you transport to the target system so that only one transformation has to be maintained in the test system. The same transformation is created automatically for each source system in the production system.

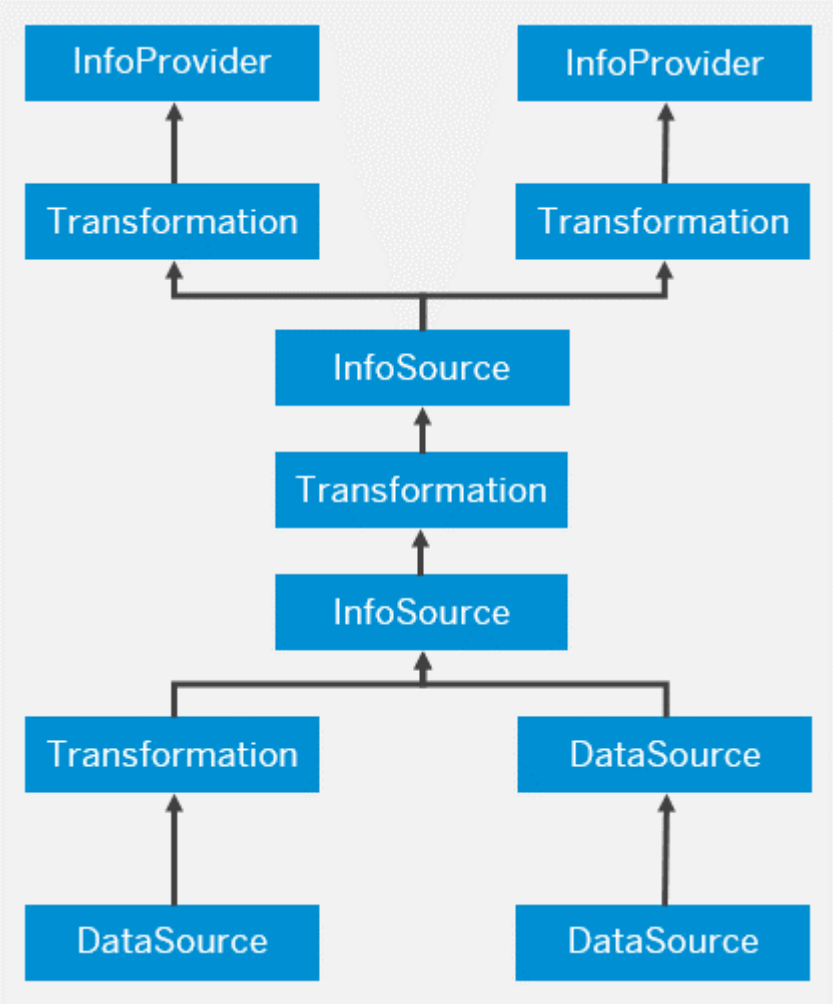
2. Data Flow with One InfoSource

The DataSource is connected to the target by means of an InfoSource. There is one transformation between the DataSource and the InfoSource and one transformation between the InfoSource and the target.



We recommend that you use an InfoSource if you want to connect a number of different DataSources to a target and the different DataSources have the same business rules. In the transformation, you can align the format of the data in DataSource with the format of the data in the InfoSource. The required business rules are applied in the subsequent transformation between the InfoSource and the target. You can make any changes to these rules centrally in this one transformation, as required.

3. Data Flow with Two InfoSources



We recommend that you use this type of data flow if your data flow not only contains two different sources, but the data is to be written to multiple targets that are identical or almost identical. The required business rules are executed in the central transformation so that you only have to modify the one transformation in order to change the business rules. You can connect sources and targets that are independent of this transformation.

3.6.2 Authorizations for InfoSources

To work with InfoSources, you need authorization object S_RS_IOBJA.

3.6.3 Transporting InfoSources

The InfoSource is integrated into BW's TLOGO framework and can be transported.

The transport object is called TRCS.

3.7 DataStore Object

The DataStore object is the central object for data storage and data consolidation in the SAP BW/4HANA system.

The DataStore object can be used very differently in the data warehouse, depending on the properties you select. The DataStore object can contain both InfoObjects and fields. This allows you to load data into the SAP BW/4HANA without having to assign InfoObjects. Thanks to its new request management, the DataStore object is particularly well suited to deal with frequent loading and large amounts of data.

The ODP source system is available to help you update data from one InfoProvider to another in a different SAP BW/4HANA system (data mart). More Information: Transferring Data from SAP BW or SAP BW/4HANA Systems via ODP (InfoProviders, Export DataSources) [page 165]

Modeling for the DataStore object is integrated in the BW Modeling tools.

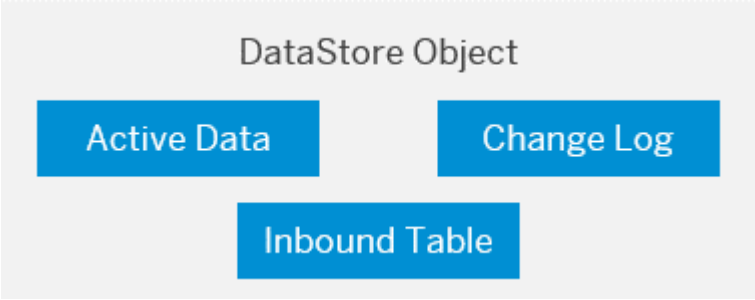
Related Information

Creating DataStore Objects [page 273]
Modeling with Fields Instead of InfoObjects [page 39]

3.7.1 Structure of the DataStore Object

The DataStore object consists of a maximum three tables: the inbound table, the change log and the table of active data.

You might not need all three tables. This depends on how you want to use the DataStore object.



The data is initially loaded into the inbound table. The data is either read from the inbound table directly, or it is activated first and then read from the table of active data. This depends on how the data is used. The change log contains the change history for the delta update from the DataStore object to other data targets.

3.7.2 Modeling Properties

The modeling properties enable you to control how you can use your DataStore object.

You can set the following modeling properties:

- *Standard DataStore Object*
  The standard DataStore object is suitable for the majority of aplication cases and for reporting. More information: Standard DataStore Object [page 71]
- *Staging DataStore Object*
  The staging DataStore object can be used in different ways by selecting its properties. More information: Staging DataStore Object [page 73]
- *Data Mart DataStore Object*
  The data mart DataStore object is optimized for reporting and analysis. More information: Data Mart DataStore object [page 76]
- *Direct Update DataStore Object*:
  In the DataStore object for direct update, you can use an API to write the data directly to the table of active data. More information: DataStore Object for Direct Update [page 78]

Special Properties

- *Inventory Enabled*: In standard DataStore objects and data mart DataStore objects, you can include non-cumulative key figures and define validity characteristics. More Information:
- *Planning Enabled*: You can enable planning for standard DataStore objects with change log, data mart DataStore objects, and direct update DataStore objects.
- *Write Interface Enabled*: Data can be moved to the inbound table using a tool such as Data Services or *Push* on SAP Cloud Platform Integration (CPI). This property can be set for all types of standard DataStore objects and staging DataStores. More Information: DataStore Object with Write Interface [page 79]

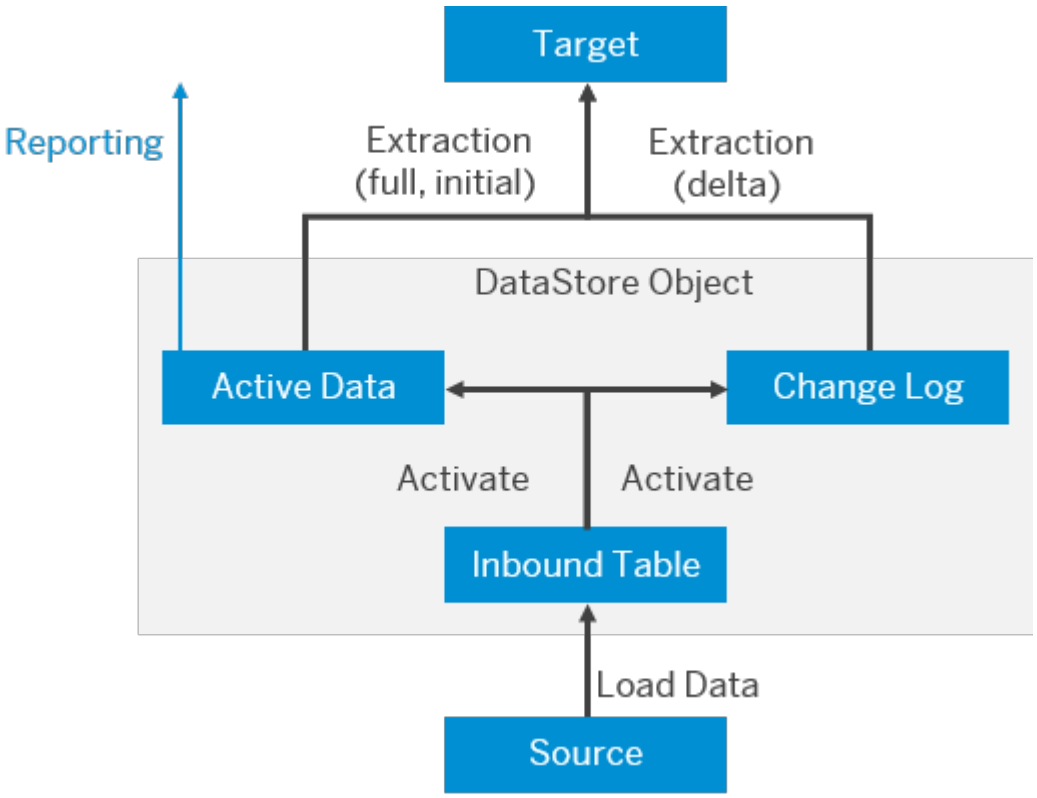
3.7.2.1 Standard DataStore Object

For the standard DataStore object, you can select the following properties:

- *Write Change Log*: If you choose this option, the delta (new, deleted, and changed records) is saved in the change log. The change log is used to extract the delta. Only if the DataStore object has a change log can requests be rolled back from the DataStore object, that is, the status before the activation of the request can be restored.
- *Snapshot Support*: If your DataSource only delivers the current dataset as a “FULL”, by setting this indicator, deleted data records can be identified and updated. Upon activation, the system recognizes records that are in the table of active data but not in the load request. These are written to the change log as reverse images.

  > **Note**
  >
  > Make sure that all records are contained during every load, as data could otherwise be lost.

- *Unique Data Records*: If you only load unique data records (data records with non-recurring key combinations) into the DataStore object, you can select this property. If the indicator is selected, during activation the system checks whether unique records exist. If a record already exists, the activation is cancelled with an error.



Load Data

Requests are loaded into the inbound table.

Activate

Activation transfers the data from the inbound table to the table of active data. Data with the same key are summarized according to the specified aggregation. If the property *Write Change Log* is selected, the difference of the activation is written to the change log.

Deletion / Rollback

If requests are not activated yet, you can delete them from the inbound table. If you want to roll back a request that has already been activated, you can do this only if you selected the property *Write Change Log*.

Extraction

Data used for the update to a further data target is extracted from the table of active data for initial and full extraction, and from the change log for delta extraction.

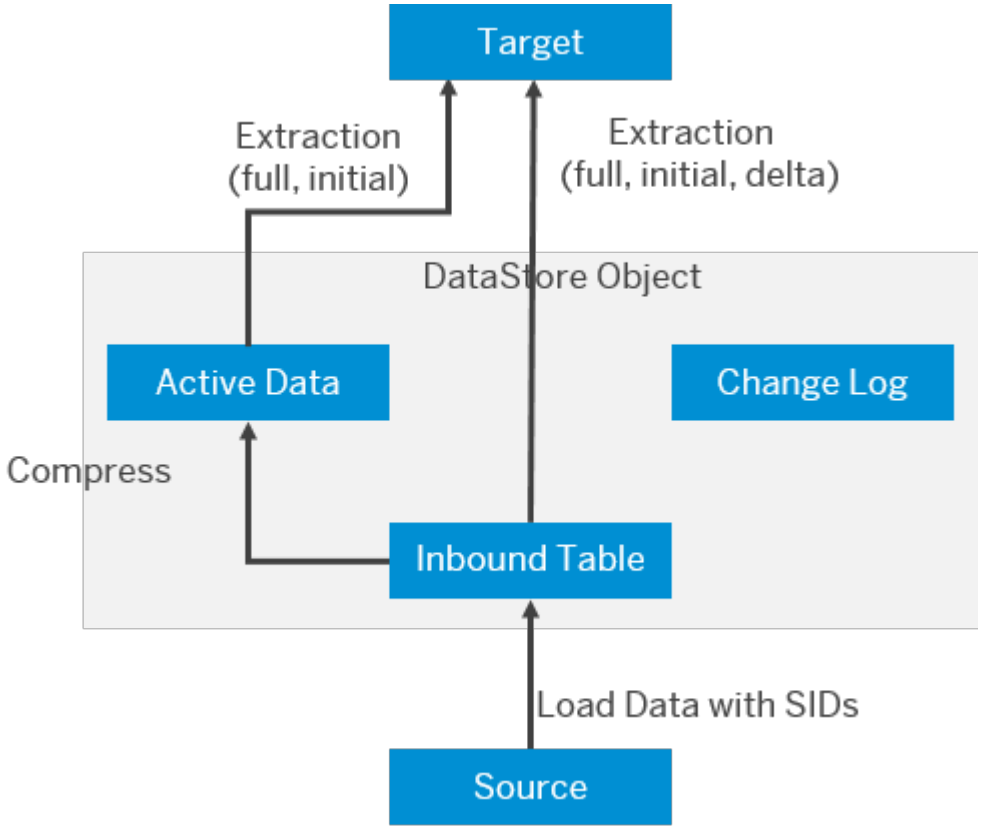
Reporting

When a query is executed, the table of active data is accessed and therefore only data that was previously activated is visible. For standard DataStore objects, there is no stable navigation reporting, as is the case with data mart DataStore objects.

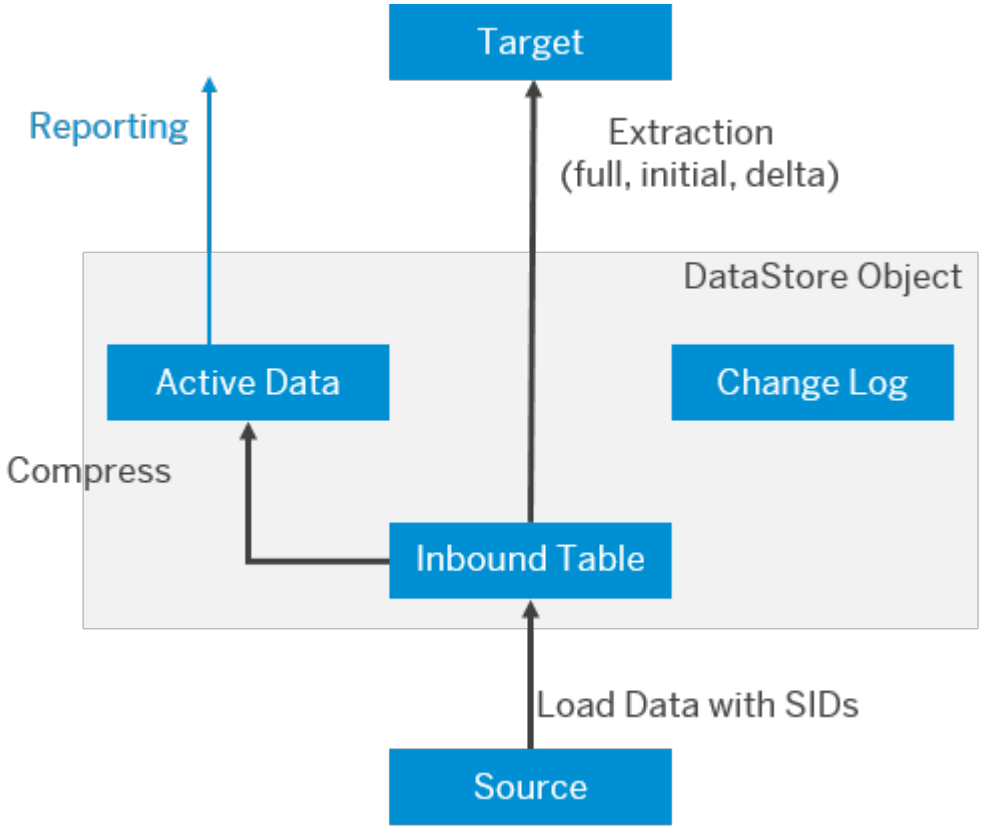
3.7.2.2 Staging DataStore Object

The staging DataStore object can be used in different ways by selecting its properties.

Structure of a staging DataStore object with the property *Compress Data*:



Structure of a staging DataStore object with the property *Reporting-Enabled*:



Load Data

Regardless of the characteristics of the DataStore object, the data is always written to the inbound table.

Activate

For DataStore objects with the properties *Reporting-Enabled* and *Compress data*, activation/compression is possible. For DataStore objects with the property *Inbound Queue Only*, it is not.

For DataStore objects with the property *Compress Data*, the term “compression” is used, since in this case the data is moved from the inbound table to the table containing the active data, and all records with the same key values are merged in accordance with the default aggregation.

For DataStore objects with the property *Reporting-Enabled* the term “activation” is used, since in this case the relevant data is replicated from the inbound table to the table containing the active data is replicated and only then is visible for reporting. Records with the same key are merged in accordance with the aggregation here too however.

Delete/Rollback

The data can only be deleted at request level if it is not contained in the inbound table yet, i.e. hasn't been activated yet.

Extraction

For DataStore objects with the properties *Inbound Queue Only* and *Reporting-Enabled*, full extraction only goes to the inbound table. For DataStore objects with the property *Compress Data* meanwhile, the inbound table and the table with the active data are accessed.

Delta extraction is always performed from the inbound table. For DataStore objects with the property *Compress Data*, a request must first be updated to all active delta DTPs before it can be compressed (activated).

Reporting

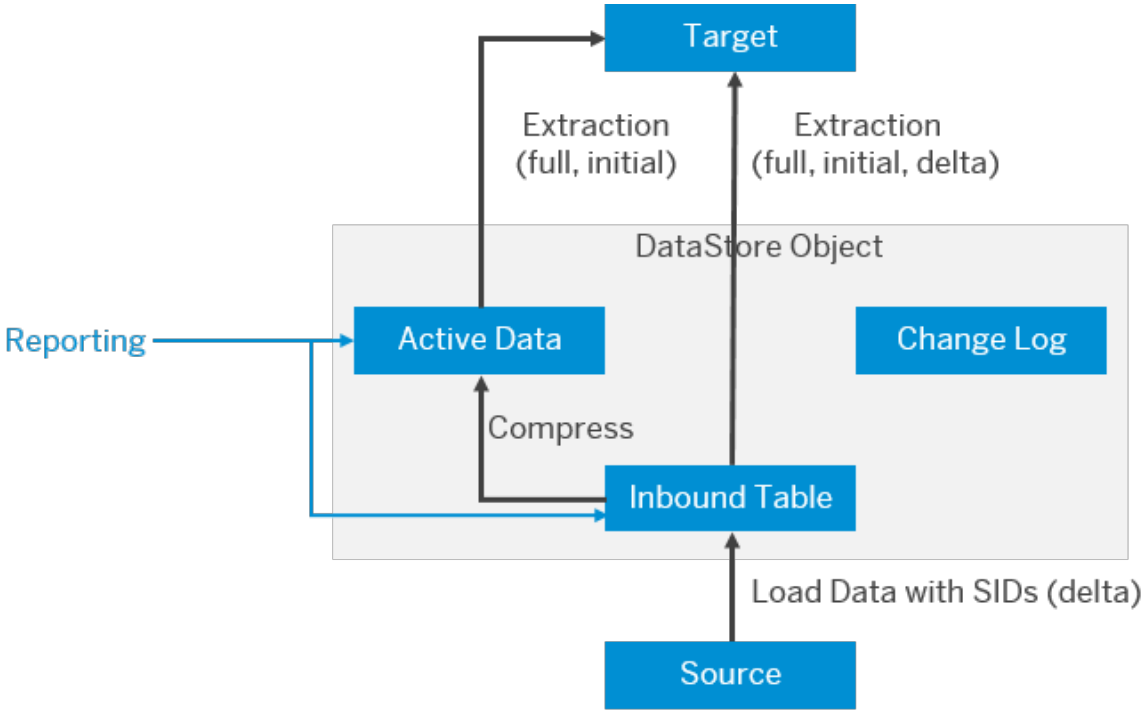
DataStore objects with the properties *Inbound Queue Only*and *Compress Data* can only be used for reporting to a very limited extent, since the data in the inbound table is only relevant logically for the object, but the query does not know the activation logic and therefore cannot aggregate over records. DataStore objects of this type can therefore not be added to a CompositeProvider.

For DataStore objects with the property *Reporting-Enabled*, reporting only takes place on the table containing the active data, meaning that the data is only visible after activation. DataStore objects of this type can be added to a CompositeProvider.

3.7.2.3 Data Mart DataStore object

The data mart DataStore object is used for reporting and analysis.

The data mart DataStore object has the following structure:



Permitted Types

In this DataStore object no InfoObjects are allowed to be used that are only display attributes. Also, no fields with the following data types are allowed to be used: STRING, RAW, RAWSTRING, DF16_RAW und DF34_RAW.

Load Data

The data is always loaded into the inbound table.

Only additive deltas can be loaded (for key figures only aggregations MAX, MIN, and SUM are allowed). Therefore, key figures cannot be overwritten.

Activate

No keys can be specified for this DataStore object type, but the records are treated as if all characteristics are in the key (logical key).

When you activate the data, the data is copied to the table of active data and grouped according to the aggregation behavior. Afterwards, the data is deleted from the inbound table. The change log is not filled.

Deletion / Rollback

For this type, requests can only be deleted if the request has not yet been activated.

Extraction

The full and initial delta extraction reads the data for the update into a further data target from the inbound table and from the table of active data.

The data for the delta extraction is read from the inbound table.

If a delta extraction is connected and active, a request can only be activated once all active delta extractors have retrieved the request.

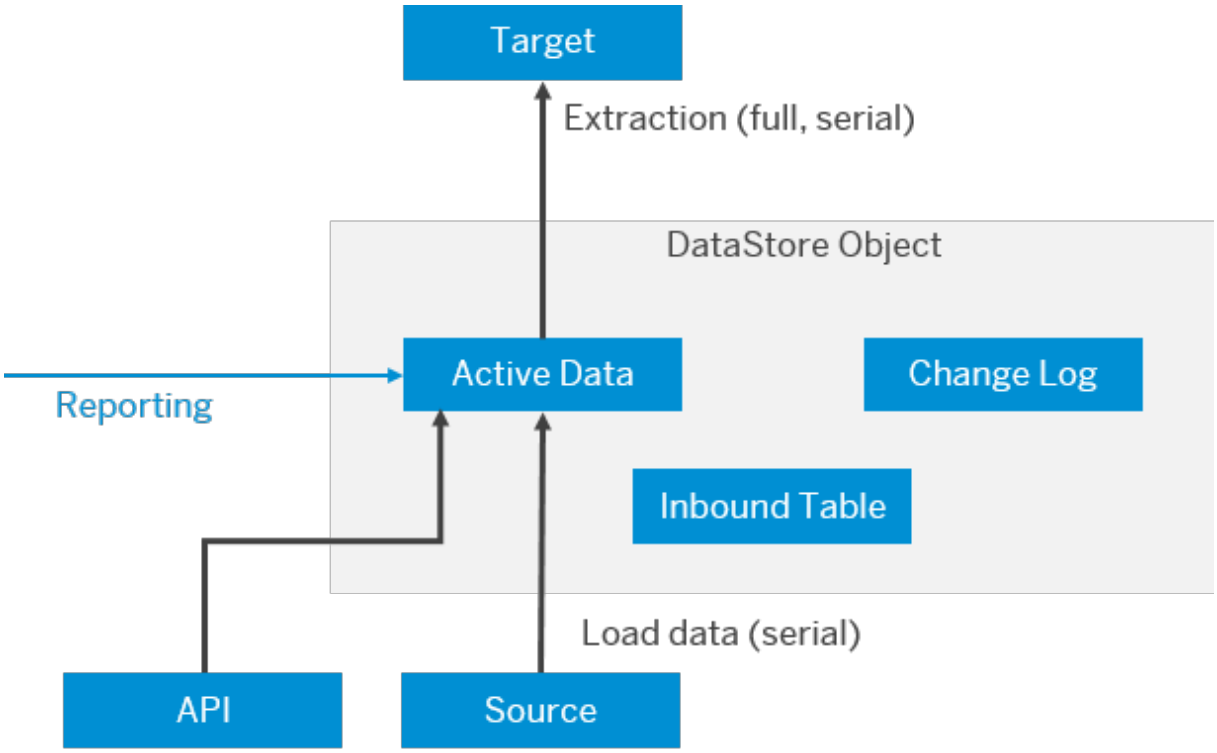
Reporting

The query reads data from the inbound table and from the table of active data. Therefore, it is not necessary to activate the data to be able to see all the data. The data read is consistent, since data is only included up to the first inconsistent request. The data read is also stable, since the quantity of data included from the DataStore object does not change during the navigation steps in the query.

3.7.2.4 DataStore Object for Direct Update

In a DataStore object for direct update, you can load the data directly into the table for active data.

The Data Mart DataStore object has the following structure:



Load Data

With this DataStore object type, the data is directly written to the table of active data.

The data is written either using a DTP or an API.

When loaded with DTP and with the API, consistency checks are made (for example, SID processing, consistency of time characteristics, areas locked by cold store or NLS).

Activate

Activation is not possible with this type.

Deletion / Rollback

Deletions can be made using requests.

Extraction

Only full extraction is possible. The data is read from the table of active data.

Reporting

All data of the active data table is visible in reporting immediately after the data is loaded.

Related Information

APIs for the DataStore Object [page 83]

3.7.2.5 DataStore Object with Write Interface

The data in this DataStore object can be moved to the inbound table using a tool such as Data Services or via SAP Cloud Platform Integration (CPI) by PUSH.

The API for writing to DataStore objects is open for third party tools and can be used by external tools. There is no certification process for this API however.

This property can be used for all types of standard DataStore object and staging DataStore object, but not together with *Inventory enabled* or *Planning enabled* .

If streaming is used to load data, the data is activated automatically.

You can use RFC or HTTP data to load data into the DataStore object. The following transfer scenarios are available. You can mix and match these as required.

Scenario	Request Administration	Effort and Complexity
Send Data Without Request (**One Step**)	A new request is opened for each data call.	Easy to implement. Suitable for scenarios with low frequency of data transfer, since otherwise too many requests can be created, thus making housekeeping necessary.

Scenario	Request Administration	Effort and Complexity
Send Data with Request (**Write into Request**)	The following sequence is used for this: *Open Request*, n times *Send Data* (with request and package), *Close Request*.	Complex implementation, since the sending system (for example the ETL tool) must manage *Open Request*, *Close Request*, and *Send Data* with a consistent ID from the data package. Suitable for scenarios with higher frequency of data transfer.
Send Data with Message Request (**Messaging Request**)	A new data package is opened internally for each data call. If no requests are opened, a request is opened when data is sent. A maximum of one request per object is open.	Easy to implement. Suitable for scenarios with higher frequency of data transfer. The request must be closed in the SAP Datasphere, SAP BW bridge system using a process chain. **Note** To do this, you need a process chain in which the *Close Mes. Requests* step (process type `PUSH_MSGR`) is scheduled prior to *inbound activation* so that the request is closed automatically. See . You can use the frequency of the process chain execution to define the number of requests, one request per hour or per day for example. **→ Tip** Note the technical specifications when using the messaging scenario: • For HTTP, pass the **`request=MESSAGING`** parameter when calling the data-Send URI. • For RFC, pass the **`MESSAGING=X`** flag when calling function module `RSO_PUSH_SEND_STRING`.

The data is transferred in the internal format.

> ### Example
>
> A date (data type DATS) is transferred as “20101231”, a time (TIMS) as “235959”.

Fields of type currency (CURR) are transferred in the standard format. During the loading process the decimal place is changed to the SAP standard format.

> ### Example
>
> 1234 JPY is transferred -> due to the currency *Japanese Yen* the value is saved in internal format as 12.34.
>
> 12.34 EUR is transferred -> due to the *Euro* currency, the value is also saved as 12.34.

On fields of type *Character* with the ALPHA conversion routine, conversion into the internal format is done automatically at loading.

> ### Example
>
> The value “1234” is transferred with the ALPHA conversion routine into a field of type *character* with length 10 The value is converted automatically to “0000001234”.

Only after-image deltas can be loaded.

You can find the URIs specific to each DataStore object to call the functions using HTTP (JSON or XML) in the BW Modeling tools, under *Properties* > *Template URIs*.

The RFC function modules are in the function group RSO_PUSH.

Related Information

SAP Note 2441826

InfoObject with Write Interface [page 59]

3.7.3 Data Tiering Properties

The data tiering properties enable you to control how you can use your DataStore object.

You can classify the data in your DataStore object as hot, warm or cold, depending on how frequently it is accessed. For more information, see .

If you have selected the temperature *Cold (External Cold Store)*, you can specify a *connection*. You can also define whether to allow exceptional updates for the cold store (*Exceptional Updates to Cold Store*). For more information, see .

3.7.4 Currency Translation and Unit Conversion

If you want to perform a currency translation or unit conversion in your DataStore object (advanced), there are various ways to model this.

InfoObject-Based

If you model your DataStore object with InfoObjects, the reference between the key figures and the associated currencies or units is guaranteed due to the InfoObject.

This is also recognized by a CompositeProvider, which is defined on the DataStore object.

Only Field-Based

If you only model your DataStore object using fields, then the key figures and currencies or units of measure can be stored in a field of any type.

If the types are reporting-compatible, then the fields are visible in a CompositeProvider that is defined on a DataStore object. This means:

A field with the data type quantity (QUAN) must have the following properties:

- The length must be > 1 and < 31 and the decimal places >= 0.
- An aggregation behavior must be set.
- It must reference a field of type UNIT. This field must have length 3 and cannot have any decimal places.

A field with the data type currency (CURR) must have the following properties:

- The length must be > 1 and < 31 and the decimal places >= 0.
- An aggregation behavior must be set.
- It must reference a field of type CUKY. This field must have length 5 and cannot have any decimal places.

If the types are not reporting-compatible, then you need to create an Open ODS view on the DataStore object, in order to perform the assignment.

Then you can create associations in the CompositeProvider. This allows you to create an assignment between the key figure and currency/unit of measure field.

Field-Based with Metadata

If you only model your DataStore object using fields, then you can assign the type CURR to the key figure, for which you want perform a currency translation, and assign the type CUKY to the currency. You can assign the type QUAN to the key figure, for which you want to perform a unit conversion, and assign the type UNIT to the unit of measure. You need to specify a field of type CUKY on the key figure.

An association with 0CURRENCY is added for the currency field in the BW back end.

The relationship between the key figure and the currency is then stored in the metadata and a CompositeProvider defined on the DataStore object can derive the assignment from the relationship.

3.7.5 APIs for the DataStore Object

You can use various APIs to load, activate and delete data.

The following APIs are available for a standard DataStore object with the property *Write Change Log*:

Type	Target Table	API for Local Use	API for RFC Use	Description
Load	Inbound Table	RSDSO_WRITE_API	RSDSO_WRITE_API_RFC	Loads data from an internal table to the inbound table. The activation can then be started.
Activate	Active Data	RSDSO_ACTIVATE_REQ_API_RFC	RSDSO_ACTIVATE_REQ_API_RFC	Activates loaded requests

The following APIs are available for a DataStore object for direct updates (Direct Update DataStore object):

Type	Target Table	API for Local Use	API for RFC Use	Description
Load	Active Data	RSDSO_DU_WRITE_API	RSDSO_DU_WRITE_API_RFC	Loads data from an internal table into the active data table.
Delete	Active Data	RSDSO_DU_DELETE_API_RFC	RSDSO_DU_DELETE_API_RFC	Deletes data from the table of active data. The table of active data can be truncated or selectively deleted.
Status	Active Data	RSDSO_DU_CLEANUP_API_RFC	RSDSO_DU_CLEANUP_API_RFC	Deletes API requests with errors. Red requests block further load requests by DTP or by API.

3.7.6 Using the Concept of Non-Active Data

If DataStore objects **do not use** the warm store, the concept of non-active data is applied for the inbound table and the change log of the DataStore objects. An exception to this rule applies in the case of inbound table of Data Mart DataStore objects. When these DataStore objects are activated, this means that the inbound table and the change log are automatically flagged for prioritized displacement from the SAP HANA main memory.

> **Note**
>
> Note that with data tiering optimization a central solution for handling and storing data in accordance with its temperature classification is available. Within the framework of this solution warm data in a scale-out landscape can be moved to a warm store (extension nodes). If a DataStore object uses the warm store, the concept of active data with preferential displacement from the main memory **cannot** be used.
>
> For this reason, the following applies: Automatic suppression for the inbound table and the change log of the DataStore objects is only performed if DataStore objects **do not use** the warm store. An exception to this rule applies in the case of inbound table of Data Mart DataStore objects.

Non-active data is data that is not required, or rarely required, in Data Warehouse processes or for analysis purposes. According to multi-temperature classification, this data is qualified as warm. This data needs to be retained in the system for various reasons (for example, storage of historical data, data storage due to legal reasons, data storage for security reasons to enable reporting-relevant layers to be recovered if necessary). The concept of non-active data allows you to make more effective use of the main memory in SAP HANA in specific cases.

This concept is based on the displacement concept for main memory bottlenecks in the SAP HANA database. If bottlenecks occur in the main memory of the SAP HANA database (or while cleaning up the main memory), strict time-based criteria are used to displace the data from the main memory into the SAP HANA file system. The data is displaced if a threshold value is exceeded by a database process requiring main memory. The columns of table partitions are displaced based on the last access time (last recently used concept). Table columns containing data that has not been accessed for the longest time are displaced first.

The SAP BW/4HANA non-active data concept ensures for specific cases that any BW data, which does not need to be actively retained in the main memory, is displaced first if bottlenecks occur. Non-active data in a SAP BW/4HANA system is handled as follows:

- If the main memory has sufficient capacity, non-active data resides in the main memory.
- If main memory bottlenecks occur, non-active data is displaced with a higher priority than other data from the main memory of the database in the SAP HANA file system.
- If non-active data that resides in the main memory is accessed, only the smallest possible volume of data is loaded into the main memory (the columns of the relevant partitions).

Reporting-relevant and process-relevant data nearly always resides in the main memory, which guarantees improved access to this data. It is also possible to reduce the main memory if a large proportion of the data in the system is non-active.

Application in SAP BW/4HANA

> **Note**
>
> Note that with data tiering optimization a central solution for handling and storing data in accordance with its temperature classification is available. Within the framework of this solution warm data in a scale-out landscape can be moved to a warm store (extension nodes). If a DataStore object uses the warm store, the concept of active data with preferential displacement from the main memory is **not** used for the tables of this DataStore object.

What Data in SAP BW/4HANAIs Seen as Non-Active Data

Most of the non-active data (classified as "warm" based on multi-temperature classification) is stored in staging DataStore objects , which only consist of one inbound table. These DataStore objects are used in the data acquisition layer. Data in the corporate memory is non-active data. Data in these layers is retained for security reasons to enable reporting-relevant layers to be restored or to guarantee access to old data. DataStore objects of these layers often contain data that is no longer required. New data is also loaded into these objects every day. Apart from the current request however, this data is usually not required for processing in SAP BW/4HANA. If a memory bottleneck occurs, these objects therefore should be displaced (instead of data for reporting-relevant objects for example) - even if this data was last used less than 24 hours ago. The data that is no longer needed should not be reloaded into the main memory when new data is loaded either.

This means that when DataStore objects that do not use a warm store are activated, the inbound table and the change log are used automatically for the prioritized displacement (with the exception of the inbound table of Data Mart DataStore objects).

Request by Request Partitioning and Displacement

DataStore objects are created in the database in request partitions. Partitioning by request means that a complete request is written to a partition. If a threshold value is exceeded, a new partition is created. The data of the next request is written to this new partition.

The Data Warehouse processes (load data, read data) usually only access data from the newest partitions. These processes always specify the partition ID for operations performed on tables. This ensures that no data from other partitions is accessed.

The fact that DataStore objects can be partitioned by request means that only the partitions containing non-active data in old requests are displaced from the main memory. The new partitions with requests relevant to the Data Warehouse processes are not displaced. These partitions are used for write and merge operations and therefore remain in the main memory.

Implementing the Concept of Non-Active Data in SAP BW/4HANA

The *Early Unload* setting is used to control the displacement of SAP BW/4HANA object data from the main memory. If a memory bottleneck occurs, the data of these objects is displaced with increased priority from the main memory (`UNLOAD PRIORITY7`). This is achieved by multiplying the time elapsed since the object was last accessed by a specific factor. This means that objects flagged with *Early Unload* are displaced more quickly than objects, which have not been accessed for a long time (and this setting has not been applied to these objects).

The inbound table and the change log of DataStore objects that do not use a warm store are automatically assigned the *Early Unload* setting (with the exception of the inbound table of Data Mart DataStore objects. You cannot change this setting.

The fact that DataStore objects are partitioned by request has the following effects:

- The columns of a non-active partition of a DataStore object are displaced with top priority from the main memory.
- Once a partition has been displaced, it is not reloaded into the main memory because new data is only written to the newest partition.
- Older data is usually not accessed. If older data is accessed, it is loaded into the main memory.
  This usually occurs with loading processes used to create new targets or when data has to be reloaded. In these cases, it is justifiable to initially load the data into the main memory.

- If not all the columns of displaced objects are accessed (for example, transformations that only read specific columns), only these specific columns are loaded into the main memory. All other columns remain in the SAP HANA file system.

Notes and Restrictions

- Note that displacement from the main memory is based on actual access and is **not** based on the data classification for access frequency (warm). If warm data is frequently needed, this data is therefore **not** displaced from the main memory (as explained above).
- The system is optimized in such a way that DataStore objects can only be accessed with the relevant partition ID. This avoids the need to load the entire table into the main memory.

> ⚠ Caution
>
> Avoid accessing these tables. In other words, avoid making regular manual changes (SQL Editor, transaction SE16) or using customized code. Otherwise, the entire table will be loaded into the main memory.

Monitoring the Handling of Non-Active Data

You are in the *Non-Active Data Monitor in the SAP HANA Database* (transaction `RSHDBMON`)).

> ⓘ Note
>
> You can call also call the monitor from the Data Warehousing Workbench (transaction `RSA1`). Choose *Administration* > *Monitors* > *Active/Non-Active Data*.

Initially, the upper part of the screen provides you with information about the system, while the lower part contains a table with information about the assigned and used memory consumption of the various services of the host.

Setting the search criteria for the detailed selection

1. To view information about the handling of non-active data belonging to a given object, choose . The *Detailed Selection* dialog box appears.
2. Restrict the **search** to the required object type. You can select from the following: *InfoArea*, *InfoProvider*, *DataSource* and the more runtime intensive options *All Objects* and *All Objects with Unloads*.

   > ⓘ Note
   >
   > All selections flagged with *Runtime Warning* can entail very long runtimes, especially when run for the first time. Subsequently, these operations are faster, as the relevant accesses are buffered.
3. Enter the technical name of the object. Input help is provided.
4. Define the search method:
   - *Exact Match*: The system searches for the exact technical name of the object of the specified object type entered in the fields described above.
   - *Search Template*: The system uses the entries made as a search template for the analysis.

5. Define which of the object's tables to search for. You have the following options:
   - Only tables with the setting "Early Unload"
   - Only tables without the setting "Early Unload"
   - If both options are selected, the system searches for all tables. This is the default setting.

> **Note**
>
> Note that these settings function as a filter on all other windows.

6. Make other **settings**:
   - *Unload Information in Header Area (Warning: Runtime)* : If you choose this option, the following unload information is displayed in the header area:
     - *Total Unloads via Partitions since* the time selected in the next step: Suppression is performed at column level. There are x partitions, of which suppressions are performed y times. The *Total Unloads via Partitions* indicates the maximum suppressed columns per partition.

       > **Example**
       >
       > Data has been suppressed once from partition 3 and three times from partition 5. *Total Unloads via Partitions* would be four in this case, as data has been suppressed from all partitions in the system four times.

     - *Number of these that are unloads on reporting/staging*partitions
     - *Number of Tables Affected*
     - *Number of these that are reporting/staging tables*
   - *Unloads since*: You can enter an appropriate point in time here.
   - *No Refresh During Navigation*: This setting is selected by default in order to keep the runtime short. It means however that you do not see changes made to the data during analysis.

     > **Note**
     >
     > If required, you can subsequently update the data display at any time by choosing .

7. Choose . In the upper screen area, the monitor shows you the unload information if you have selected this option. In the lower screen area it contains a table with detailed information about handling of non-active data in the SAP HANA database, filtered to the object set in your search query.

Analysis of Displayed Data

Depending on the search query, you see the tables of objects belonging to an InfoArea, InfoProvider or DataSource. By clicking on the underlined information you can navigate to further details:

- Column *InfoProvider*: If you click on the technical name of an InfoProvider or a Data Source, the system displays the details for all tables participating in this object and all partitions.
- Column *Unloads*: If you click on the number of unloads for a given object, the system displays the details for all affected partitions (partition ID, host, early unload, unloads, unload reason).

> **Caution**
>
> Note the following exception: If the system shows `-1` as the partition ID, this means that the entire table has been suppressed from the main memory (for example with unload reason `EXPLICIT`, meaning manual unload).

- Column *Available in Memory?*: : You have the following options:

Status of Availability in the Main Memory	Description
NO	The table is not available in the main memory.
FULL	The entire table is available in the main memory. If you click on status FULL for a table, see the details such all entries for the memory, the number of records and the last access time.
PARTIALLY	Parts of the table are not available in the main memory. To be able to tell which one it is, you can click on the PARTIALLY status for the relevant table. You then see all columns of a partition. If columns of a partition are missing, these are **not** marked in the *Available in Memory?* column. If just an index is missing however, the columns in the *Available in Memory?* column are marked, as indexes are not listed separately.

If you click on the entry of the unloads for a given object, the system displays details for unloads of this object.

More Information

- For more information about classification of data based on access frequency, see .
- For information about data tiering optimization, see.
- For information about main memory consumption in the SAP HANA database, see the SAP HANA Administration Guide at https://help.sap.com/viewer/p/SAP_HANA_PLATFORM.
- For information about how the non-active data concept affects sizing, see SAP Note 2296290 (New Sizing Report for BW on HANA).

3.7.7 External SAP HANA SQL View

The external SAP HANA SQL view is a stable SAP HANA view which you can use to access the data of a DataStore object directly.

When you activate a DataStore object, an External SAP HANA SQL view is generated. This view is displayed in the definition of the DataStore object in the screen area below the editor.

It is a technical API that can be called in programs, for example in ABAP and SAP HANA SQLScript in routines for transformations. For more complex data warehousing operations (such as delta extractions), you should use the open hub destination or OData, and not this API.

An entry is added in the ABAP Dictionary for the External SAP HANA SQL view. The calculation formula is defined when the DataStore object is activated and is dependent on the settings. For this reason, you are not allowed to make any changes in the ABAP Dictionary.

The view is assigned a name according to the following model: /<namespace>/A<name of the DataStore object>8.

The view contains the following fields:

- All physically existing semantic fields (including non-reporting-relevant fields)
- The marker for non-cumulative key figures
- Technical field COLD_STORE_FLAG
- The technical fields REQTSN, DATAPAKID, RECORD and RECORDMODE for DataStore objects of the types Staging DataStore object with the property *compress data* or *reporting enabled* and *data mart DataStore object*
- Technical field RECORDTP is included only if the DataStore object contains at least one non-cumulative key figure.

The view does not contain the following fields:

- Virtual key figures such as stock coverage and referencing key figures

Name and semantics of the fields:

- By default, the field names of InfoObjects and fields are used.
- Two columns are generated for key figures of type “amount” (even if the DataStore object only contains one column for it):
  - A column named <field name>__INT that contains the values in internal format (without decimal point shifting)
  - A column named <field name>__EXT that contains the values in external format (with decimal point shifting or fixed currency)

  If the field name is too long after the suffix is added, then a GUID is used instead of the original field name. For example: IAZO5I2O5O7JQGWMOV2670QI3__EXT and IAZO5I2O5O7JQGWMOV2670QI3__INT. The GUID is stored in table RSOADSOFIELDMAP and is also used for reporting.
- For non-cumulative key figures, an additional column named <field name>__NCUM_REFP ' is generated. If recordtp = 1 for a request,
  - the value of the corresponding inflow column (or delta column) is copied to this column
  - the corresponding inflow column (or delta column) is deleted
  - the values of all SAP time characteristics (0CAL*, 0FISC*) are set to their maximum value.

  If recordtp < > 1 for a request,
  - this column is initial.

  For non-cumulative key figures of type *amount* the suffixes are concatenated: __NCUM_REFP_INT and __NCUM_REFP_EXT.
- For all requests that read from the cold store/NLS, an X is set in the technical column COLD_STORE_FLAG. For all other requests, this column is empty. This column can be used as a filter. For example, if COLD_STORE_FLAG = '', the SQL optimizer should use pruning to access the cold store.

Tables and Keys

Depending on the type of DataStore object the External SAP HANA SQL view contains different tables and has different keys.

For standard DataStore objects with or without a change log, and staging DataStore objects with the property *Inbound Queue Only*:

Tables	Keys
Table of active data	Fields of the semantic key

For Data Mart DataStore objects:

Tables	Keys
Inbound table (all green requests up to the first red or yellow request), table of active data	REQTSN, DATAPAKID, RECORD, COLD_STORE_FLAG, all characteristics (if there are less than 251), or only the first 251 characteristics if there are more than 251 (Fields REQTSN, DATAPAKID, RECORD, are for all requests read from the table of active data, initial)

For write-optimized DataStore objects:

Tables	Keys
Inbound table (all green requests up to the first red or yellow request)	REQTSN, DATAPAKID, RECORD, COLD_STORE_FLAG

For Staging DataStore objects with the property *Compress Data*:

Tables	Keys
Inbound table (all green requests up to the first red or yellow request), table of active data	REQTSN, DATAPAKID, RECORD, COLD_STORE_FLAG (Fields REQTSN, DATAPAKID, RECORD are generated during running operation if they are read from the table of active data)

For Staging DataStore objects with the attribute *Reporting Enabled*:

Tables	Keys
Table of active data	Fields of the semantic key

If a cold store/nearline storage exists, the ON table (virtual table used for access to the archive) is added to the view. If the DataStore object contains non-cumulative key figures, the RP (reference point) table is added to the view.

Repair

You can repair the view using reports RSDG_ADSO_ACTIVATE and RSDG_ADSO_ACTIVATE_ALL. For existing DataStore objects, for which you also want to use this feature, you can generate the External SAP HANA SQL view later using report RSDG_ADSO_ACTIVATE.

3.7.8 Transporting DataStore Objects

The DataStore object is integrated into the TLOGO framework of the SAP BW/4HANA system and can be transported.

The transport object is called ADSO.

3.8 Semantic Groups

A semantic group is used to generate semantically partitioned data models.

The semantic group is not an InfoProvider in its own right. It is a tool that used to generate a group from semantically related DataStore objects (advanced). To simplify reporting and analysis over all components of a semantic group, a CompositeProvider can be generated.

Using semantically partitioned data models groups offer the following advantages when compared with individual DataStore objects (advanced):

- Better performance with mass data: Semantic partitioning means that the data sets are distributed over several data containers. This means that runtimes are kept short even if the data volume is large.
- Isolation of the data: Error handling is better. If a request for a region ends with an error, for example, the entire InfoProvider is unavailable for analysis and reporting. With a semantically partitioned data model, the separation of the regions into different partitions means that only the region that caused the error is unavailable for data analysis.
- Working with different time zones: EDW scenarios usually involve several time zones. With a semantically partitioned data model, the time zones can be separated by the partitions. Data loading and administrative tasks can therefore be scheduled independently of the time zone.

Using SAP HANA means that the semantic group has a smaller number of components that contain a large amount of data, as the performance is greatly improved by SAP HANA.

You can generate new components from a semantic group. The structure is taken over from a DataStore object (advanced) template. Existing DataStore objects (advanced) can also be added to the group if certain conditions are met. The DataStore objects (advanced) remain objects in their own right.

Staging

When loading data to a DataStore object (advanced) that is part of a semantic group, a filter in the DTP stipulates that only data that is suitable for the DataStore object as part of the semantic group is extracted. The DTP is given the status *green* even if records have been filtered out.

Reporting and analysis on a semantic group

When creating the semantic group, you can generate a CompositeProvider that contains all DataStore objects (advanced) in the semantic group and assigns them to one another by union. This provides you with the object

for reporting and analysis. This CompositeProvider is overwritten every time a change is made to the semantic group. It should therefore not be changed manually.

3.8.1 Authorizations for Semantic Groups

There is a specific authorization object for working with semantic groups. The authorization is covered by the objects to be created (DataStore object (advanced) and CompositeProvider).

3.8.2 Transporting Semantic Groups

The semantic group itself cannot be transported.

In the development system, objects are generated from the semantic group. These objects can be transported to the production system.

For the transport, the following conditions apply:

- All objects are in the same package.
- All objects are on the same request.

3.9 CompositeProvider

In a CompositeProvider, you can merge data from BW InfoProviders with data from SAP HANA views using union and join, or just merge data from BW InfoProviders and SAP HANA views.

The SAP HANA views used can be in HDI containers provided that you have activated consumption of SAP HANA views in HDI containers under *SAP BW/4HANA Customizing Implementation Guide* > *SAP BW/4HANA* > *Data Modeling* > *Settings for Consumption of SAP HANA Views* in Customizing.

The CompositeProvider's join and union operations are performed in SAP HANA.

You can run queries on a CompositeProvider just like on other BW InfoProviders. You acquire SQL access to the data if you generate a SAP HANA view from the CompositeProvider.

Delta extraction by DTP is supported for CompositeProviders with union, which are based on data mart DataStore objects.

Restrictions

- Only input parameters are supported, not variables.
- There are restrictions to how CompositeProviders can be structured and nested. More Information: Supported Assignments [page 95]

Related Information

Creating CompositeProviders [page 295]
SAP Note 2271658

3.9.1 Supported Providers

The CompositeProvider offers a range of providers for modeling.

Providers Supported for Union Operations

For CompositeProviders that only contain union assignments, the following providers are supported:

- SAP HANA views (calculation views and analysis views)
- The following BW InfoProviders: DataStore object, Open ODS view, InfoObject, aggregation level

SAP HANA views generated from BW InfoProviders are not supported.

DataStore objects should have properties that are relevant for reporting. In other words, they should be of type data mart DataStore object, standard DataStore object with the property *Write Change Log* or staging DataStore object with the property *Reporting Enabled*.

Open ODS Views that you want to use in CompositeProviders can only contain fields with InfoObject-compatible data types. These are the following data types: CHAR, CUKY, CURR, DEC, DATS, FLTP, INT4, NUMC, TIMS, QUAN, UNIT. Data type STRG is also supported for particularly long characteristic-type field values.

Supported providers that contain more than just a union

For CompositeProviders that contain both join and union assignments, the following providers are supported:

- SAP HANA views (calculation views and analysis views)
- The following BW InfoProviders: DataStore object, OpenODSView (with CalcScenario Used to true), InfoObject

DataStore objects should have properties that are relevant for reporting. In other words, they should be of type data mart DataStore object, standard DataStore object with the property *Write Change Log* or staging DataStore object with the property *Reporting Enabled*.

In the case of DataStore objects, data can be read from the near-line storage for unions and joins.

3.9.2 Using HDI Containers

You can use SAP HANA views in CompositeProvidern that are contained in HDI containers.

Prerequisites

You have to activate consumption of SAP HANA views in HDI containers under *SAP BW/4HANA Customizing Implementation Guide* > *SAP BW/4HANA* > *Data Modeling* > *Settings for Consumption of SAP HANA Views* in Customizing.

In terms of consumption, there is no difference between SAP HANA views in HDI containers and SAP HANA views in the repository.

SAP HANA Views in ABAP-Managed HDI Containers

If you use ABAP-managed HDI containers, the SAP HANA views are transported with the BW objects. HDI objects are transported using the The SAP HANA Transport Container (HTC).

SAP HANA Views in XSA Application-Managed HDI Containers

If you use an XSA application-managed HDI container, you have to perform a number of manual steps. This includes maintaining the logical schema mapping in the source system and the target system. The logical schema name must be the same in all systems.

- Create the logical schema names in the ABAP Development Tools.
- Maintain the logical and physical schema name in transaction DB_SCHEMA_MAP.

You also have to perform the transport manually: SAP HANA views that are contained in XSA application-managed HDI containers are not transported with the other BW objects.

Related Information

Transporting CompositeProviders [page 109]

3.9.3 Supported Assignments

The technical characteristics "Request ID" (0REQUID) and "Record Type" (0RECORDTP) cannot be assigned. They are automatically assigned at runtime. Technical characteristic 1ROWCOUNT is not displayed. If it is in one of the participating providers, it is added automatically as a key figure.

Non-cumulative key figures and other key figures with special semantics (such as elimination of internal business volume) must have a 1:1 assignment to the structure of the CompositeProvider. The *Direct Usage of Associated Object by Name* property also has to be set.

3.9.4 Temporal Join

You use a temporal join to map a period of time.

A temporal join is a join that contains at least one time-dependent characteristic or a pseudo time-dependent InfoProvider.

Related Information

Using Temporal Joins [page 301]

3.9.5 Possible Assignments of Data Types

When a field is assigned, the system checks whether the target field is suitable for the assignment being performed. A target field is suitable if the following apply:

- It has not already been assigned to another field in the source structure
- It has a compatible data type

Compatible Data Types

Source / Target	CHAR	SSTR	STRG	NUMC	CUKY	UNIT	DATS	TIMS	LANG	INT1	INT2	INT4	INT8	FLTP	DEC	D34D	D16D	QUAN	CURR	RAW16
CHAR	X	X	X	X	X	X	X	X	X											X
SSTR	X	X																		

Source / Target	CHAR	SSTR	STRG	NUMC	CUKY	UNIT	DATS	TIMS	LANG	INT1	INT2	INT4	INT8	FLTP	DEC	D34D	D16D	QUAN	CURR	RAW16
STRG	X		X																	
NUMC	X			X			X	X												
CUKY	X				X															
UNIT	X					X														
DATS	X			X			X													
TIMS	X			X				X												
LANG	X								X											
INT1										X	X	X	X							
INT2											X	X	X							
INT4												X	X							
INT8													X							
FLTP														X	X	X	X			
DEC														X	X	X	X	X	X	
D34D														X	X	X	X			
D16D														X	X	X	X			

Source / Target	CHAR	SSTR	STRG	NUMC	CUKY	UNIT	DATS	TIMS	LANG	INT1	INT2	INT4	INT8	FLTP	DEC	D34D	D16D	QUAN	CURR	RAW16
QUAN															X			X		
CURR															X				X	
RAW16	X																			X

Compatibility Between SAP HANA Primitive Types and ABAP DDIC Types

SAP HANA Primitive Type	ABAP DDIC Type
CHAR	CHAR
ALPHANUM	CHAR
BIGINT	DEC
BLOB	RSTR
CLOB	STRG
DATE	DATS
DECIMAL	DEC
DOUBLE	FLTP
FLOAT	Not supported
INTEGER	INT4
NCLOB	STRG
NVARCHAR	STRG
REAL	FLTP
SECONDDATE	DATS
SHORTTEXT	STRG
SMALLDECIMAL	DEC

SAP HANA Primitive Type	ABAP DDIC Type
SMALLINT	INT4
TEXT	STRG
TIME	TIMS
TIMESTAMP	DATS
TINYINT	INT4
VARBINARY	RSTR
VARCHAR	STRG

3.9.6 Scenarios for Using CompositeProviders in CompositeProviders

Supported Scenarios

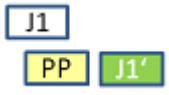
The reused CompositeProvider is included in a CompositeProvider by union and contains one union node only:



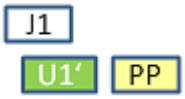
The reused CompositeProvider is included in a CompositeProvider by union and contains join assignments only (inner Join, left outer join, mixed):



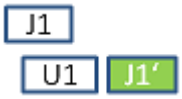
The reused CompositeProvider is included in a CompositeProvider by union and contains join assignments only (inner Join, left outer join, mixed), a further InfoProvider is included on the left:



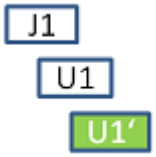
The reused CompositeProvider is included in a CompositeProvider as a left join partner and contains union nodes only:



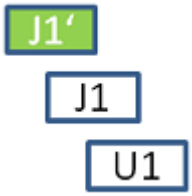
The reused CompositeProvider is included in a CompositeProvider by join and contains join assignments only (inner Join, left outer join, mixed), a further InfoProvider exists on the left:



The reused CompositeProvider is included in a union node of a CompositeProvider by union and by itself contains union node only:

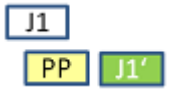


The reused CompositeProvider contains join assignments only (inner join, left outer join, mixed) and is incluced in the existing join chain of a CompositeProvider:

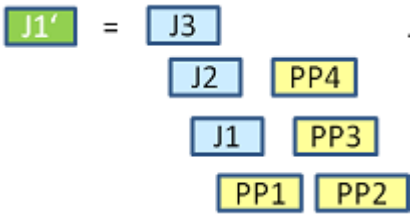


Partly Supported Scenarios

The reused CompositeProvider is included in a CompositeProvider as a right join partner and contains join nodes only:

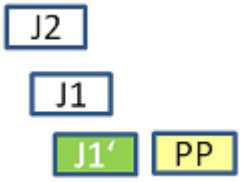


The reusable CompositeProvider could look like this (J1, J2 and J3 are left outer join nodes):

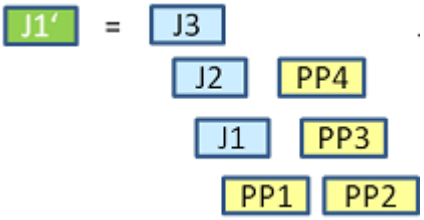


If J1 is joined with J1', the join partners for J1' can only come from PP1, the inner join part. Fields from PP2, PP3 and PP4 cannot be added to the join condition, as they come from the left outer parts.

The reused CompositeProvider J1' is inserted into the existing join chain as a left join partner.



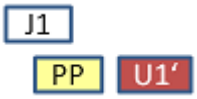
The reusable CompositeProvider could look like this (J1, J2 and J3 are left outer join nodes):



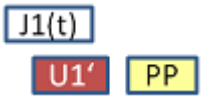
If J1 is joined with J1', the join partners for J1' can only come from PP1, the inner join part. Fields from PP2, PP3 and PP4 cannot be added to the join condition, as they come from the left outer parts.

Unsupported Scenarios

The reused CompositeProvider can never be in a join node on the right:

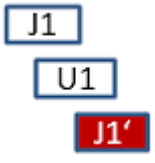


There can be no union nodes for time-dependent joins.

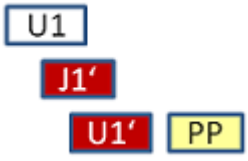




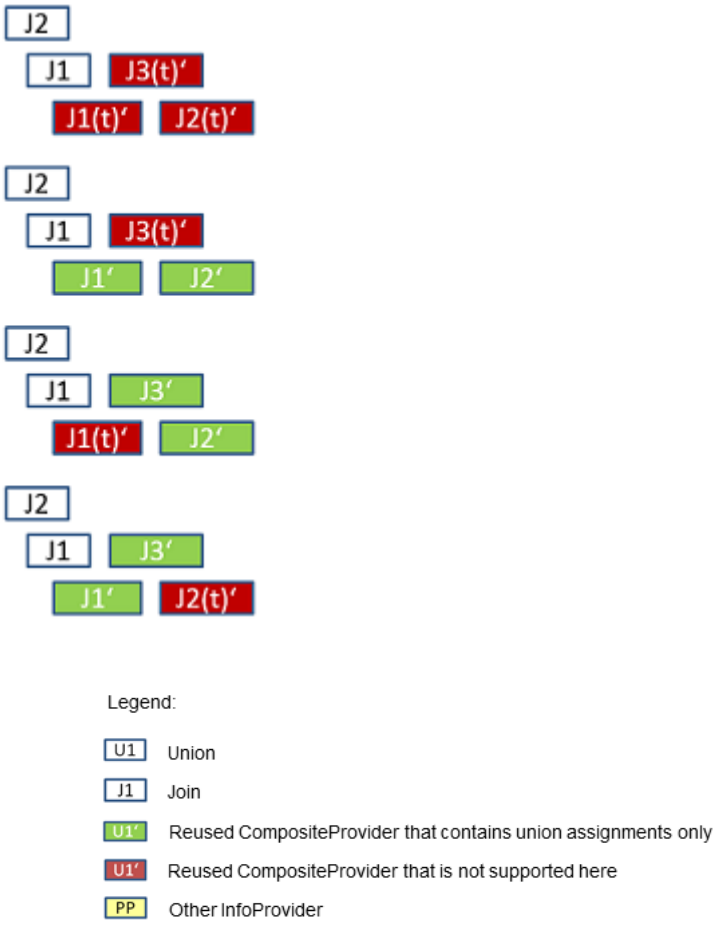
The sequence join-union-join is not allowed.



The sequence union-join-union is not allowed.



Reusable time-dependent CompositeProviders can only be added to a union node as PartProviders. For this reason, none of the following scenarios are allowed:



3.9.7 Query Properties

You can make settings for the query properties of an InfoProvider participating in a query.

The following query properties are offered:

- *Query Cache Mode*: The cache mode defines whether and how the query results and navigational states calculated by the OLAP processor should be saved as highly compressed data in a cache.
- *Cache Usage Mode*: The cache usage mode is an expert tool for the cache mode. It defines when cache entries can be used to answer a query, and when the data has to be read again.
- *Data Integrity*: You can choose which request status the data in an InfoProvider should be read with.
- *Near-Line Storage Read Mode*: You can read the data saved in a near-line storage when executing a query.

Constraints

These settings are only available for DataStore objects. The following restrictions also apply:

- For DataStore objects as right join partner, only the *Near-Line Storage Read Mode* query property is available.
- For time-dependent joins, only the *Data Integrity* query property is available.

3.9.8 Audit

The user can view the audit information for every cell of the query. This information is provided as delta values of the change history in a corresponding audit query.

Supported InfoProviders

You can use the data audit for the following InfoProviders:

- Data Mart DataStore Object: In this type of InfoProvider, the data audit is a fixed component and is always active.

> **Note**
>
> The audit characteristics should only be used for the audit function. In other words, these characteristics should not be added to InfoProviders that do not support audits.

Using Audit Characteristics in Providers

Data Mart DataStore Object

The prerequisite for this is that the *Show Audit Characteristics in Query* property is set in a CompositeProvider containing the DataStore object.

The audit infrastructure is a fixed component in the DataStore object. The audit information is updated via the request and can be included in queries via the navigation attributes.

The following InfoObjects are used:

- Time Stamp: `0REQTSN`
- User: Navigation Attribute `0REQTSN__0AUSER`
- Data source: Navigation Attribute `0REQTSN__ 0ASOURCE`

Related Information

Audit Characteristics [page 103]
Audit Query [page 104]

3.9.8.1 Audit Characteristics

The system uses specific characteristics to save audit information.

The audit characteristics are supplied with technical content from SAP BW/4HANA. These characteristics are filled by the system and cannot be modeled by users.

The following table provides an overview of the audit characteristics and their values.

Audit Characteristics

Characteristic	Values and Description
0ATIMSTMP	This characteristic contains the audit time stamp. • If the database audit is active, this characteristic contains a time stamp, which indicates when the data was saved on the database. • If the database audit is inactive, this characteristic contains a fixed time stamp, which indicates when the data audit was switched from active to inactive.
0AMODE	This characteristic contains the audit modus. The following values are permitted: • WHM: This value is used in loading mode (in other words, when data is added to the InfoProvider by loading requests). • OFF: This value is used if the data audit is set to inactive.
0AUSER	This characteristic contain the name of the user who initiated the save event. • If the data audit is active, this characteristic contains the content of the ABAP system field SY-UNAME. • If the data audit is inactive, this characteristic is empty.
0ASOURCE	This characteristic contains the name of the source of the data change at the time of saving.

Related Information

Audit [page 102]

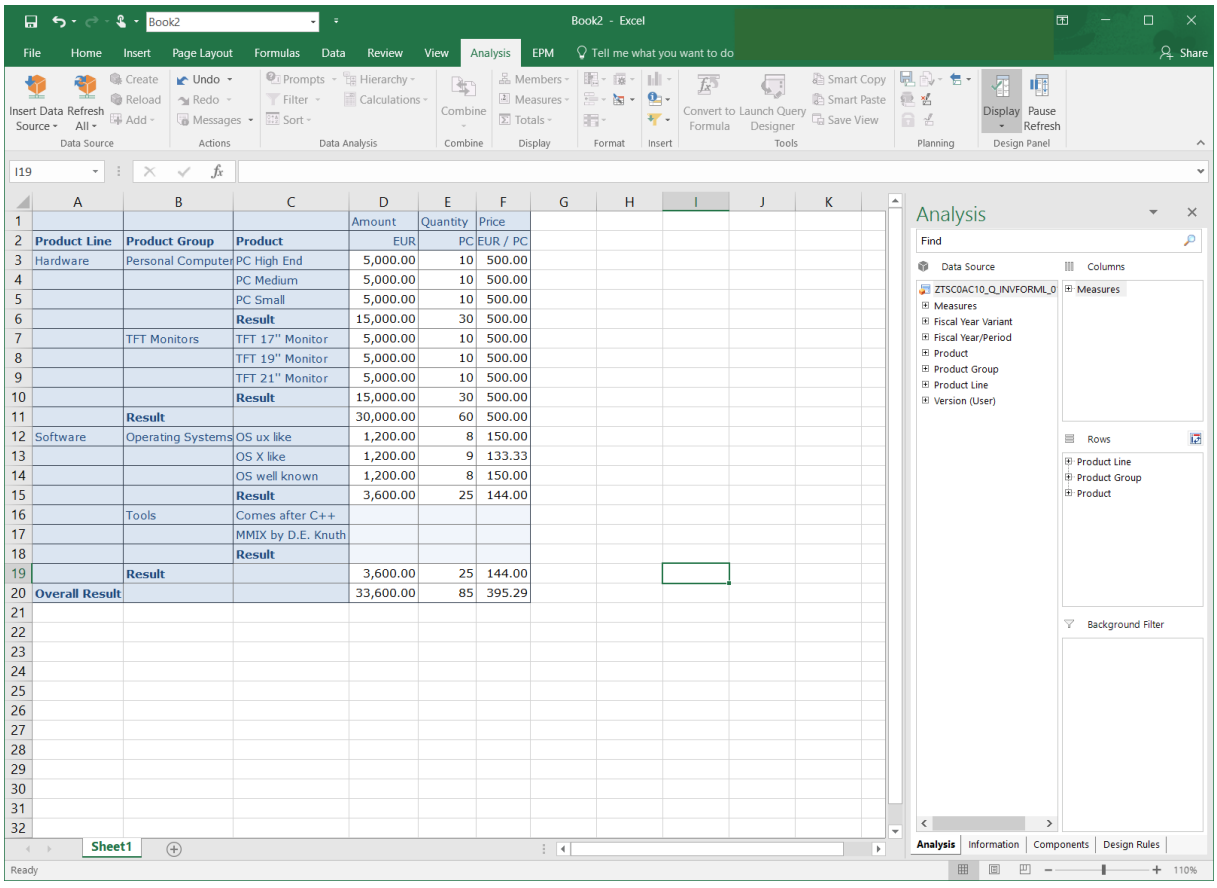
3.9.8.2 Audit Query

The following example shows how to define an input-ready query and the corresponding audit query. You can either execute an audit query in a way that allows to you to track the changes, or in a way that allows you to view the delta values.

Changes in Input-Ready Query ZTSC0AC10_Q_INVFORML_01 (With and Without Audit)

In the following example, the user has used an input-ready query to write values to a data mart DataStore object.

The following screen shows the executed query:

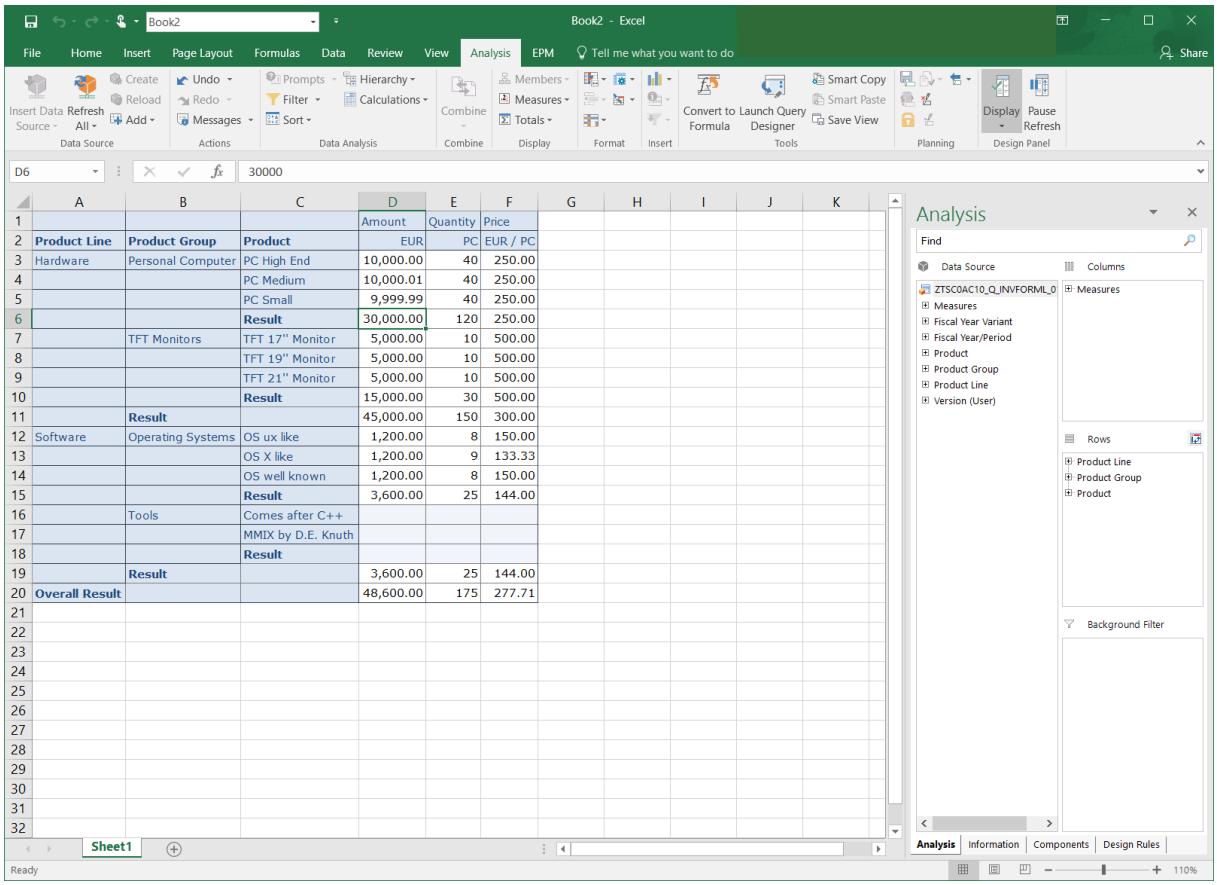


	A	B	C	D	E	F
1				Amount	Quantity	Price
2	**Product Line**	**Product Group**	**Product**	EUR	PC	EUR / PC
3	Hardware	Personal Computer	PC High End	5,000.00	10	500.00
4			PC Medium	5,000.00	10	500.00
5			PC Small	5,000.00	10	500.00
6			**Result**	15,000.00	30	500.00
7		TFT Monitors	TFT 17" Monitor	5,000.00	10	500.00
8			TFT 19" Monitor	5,000.00	10	500.00
9			TFT 21" Monitor	5,000.00	10	500.00
10			**Result**	15,000.00	30	500.00
11		**Result**		30,000.00	60	500.00
12	Software	Operating Systems	OS ux like	1,200.00	8	150.00
13			OS X like	1,200.00	9	133.33
14			OS well known	1,200.00	8	150.00
15			**Result**	3,600.00	25	144.00
16		Tools	Comes after C++			
17			MMIX by D.E. Knuth			
18			**Result**			
19		**Result**		3,600.00	25	144.00
20	**Overall Result**			33,600.00	85	395.29

Then the user made further changes in the input-ready query and then saved the changes immediately. The following changes were made:

1. Subtotal *Hardware*, *Personal Computer*: the value of *Sales Quantity* was changed from 30 to 60.
2. Subtotal *Hardware*, *Personal Computer*: the value of *Sales* was changed from 15,000 to 20,000.
3. Subtotal *Hardware*, *Personal Computer*: the value of *Sales Quantity* was changed from 60 to 120 and the value of *Sales* was changed from 20,000 to 30,000.

The following screen shows the executed query after the changes:



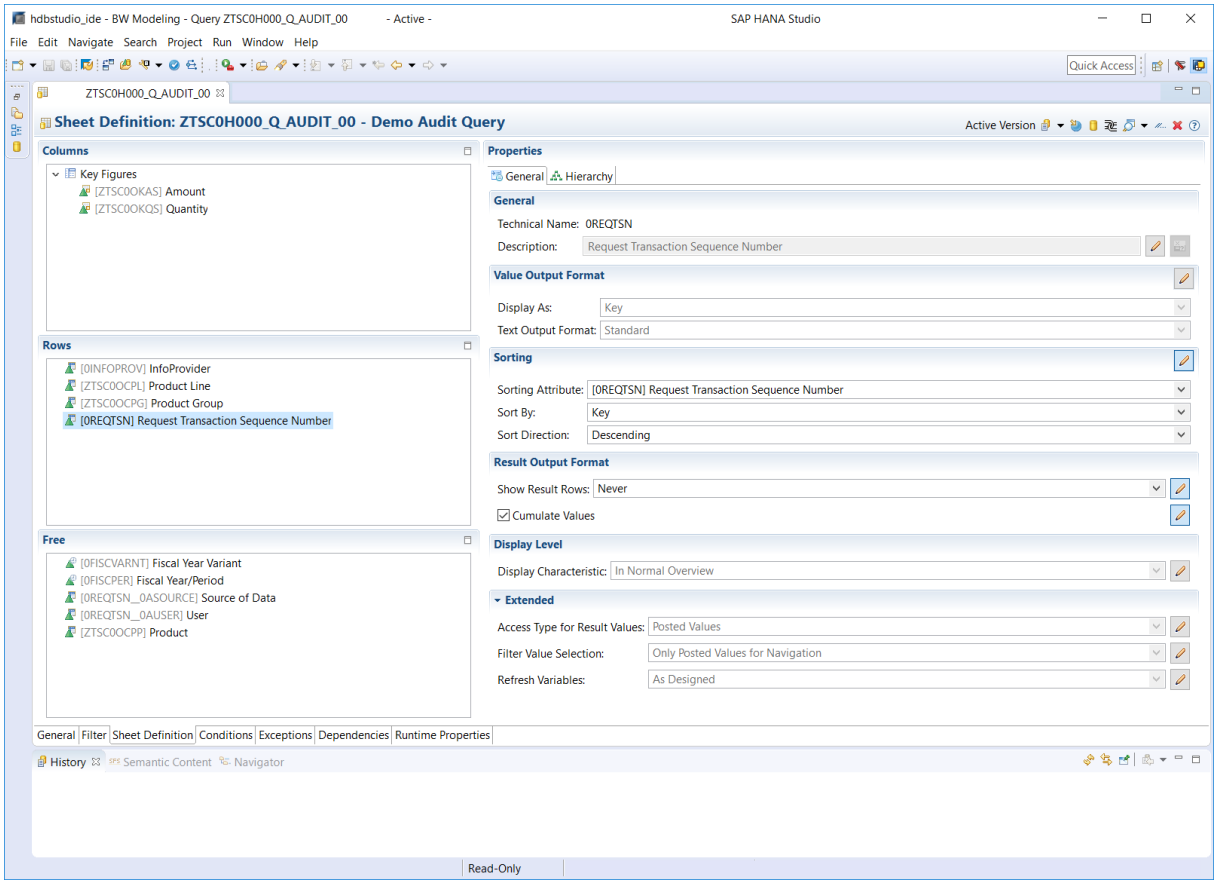
	A	B	C	D	E	F
1				Amount	Quantity	Price
2	**Product Line**	**Product Group**	**Product**	EUR	PC	EUR / PC
3	Hardware	Personal Computer	PC High End	10,000.00	40	250.00
4			PC Medium	10,000.01	40	250.00
5			PC Small	9,999.99	40	250.00
6			**Result**	30,000.00	120	250.00
7		TFT Monitors	TFT 17" Monitor	5,000.00	10	500.00
8			TFT 19" Monitor	5,000.00	10	500.00
9			TFT 21" Monitor	5,000.00	10	500.00
10			**Result**	15,000.00	30	500.00
11		**Result**		45,000.00	150	300.00
12	Software	Operating Systems	OS ux like	1,200.00	8	150.00
13			OS X like	1,200.00	9	133.33
14			OS well known	1,200.00	8	150.00
15			**Result**	3,600.00	25	144.00
16		Tools	Comes after C++			
17			MMIX by D.E. Knuth			
18			**Result**			
19		**Result**		3,600.00	25	144.00
20	**Overall Result**			48,600.00	175	277.71

Definition of Audit Query ZTSC0H000_Q_AUDIT_00

You usually create the audit query on a CompositeProvider containing all base InfoProviders that are changed with input-ready queries or planning functions. We recommend that you build the audit query in the same way as the input-ready query. However, all the characteristics of the audit dimension should be available as free characteristics.

In our example, the characteristic *Request Transaction Sequence Number* (`0REQTSN`) is drilled down in the rows. The characteristic *Product* is removed from the rows, which makes it easier to track changes to the subtotal.

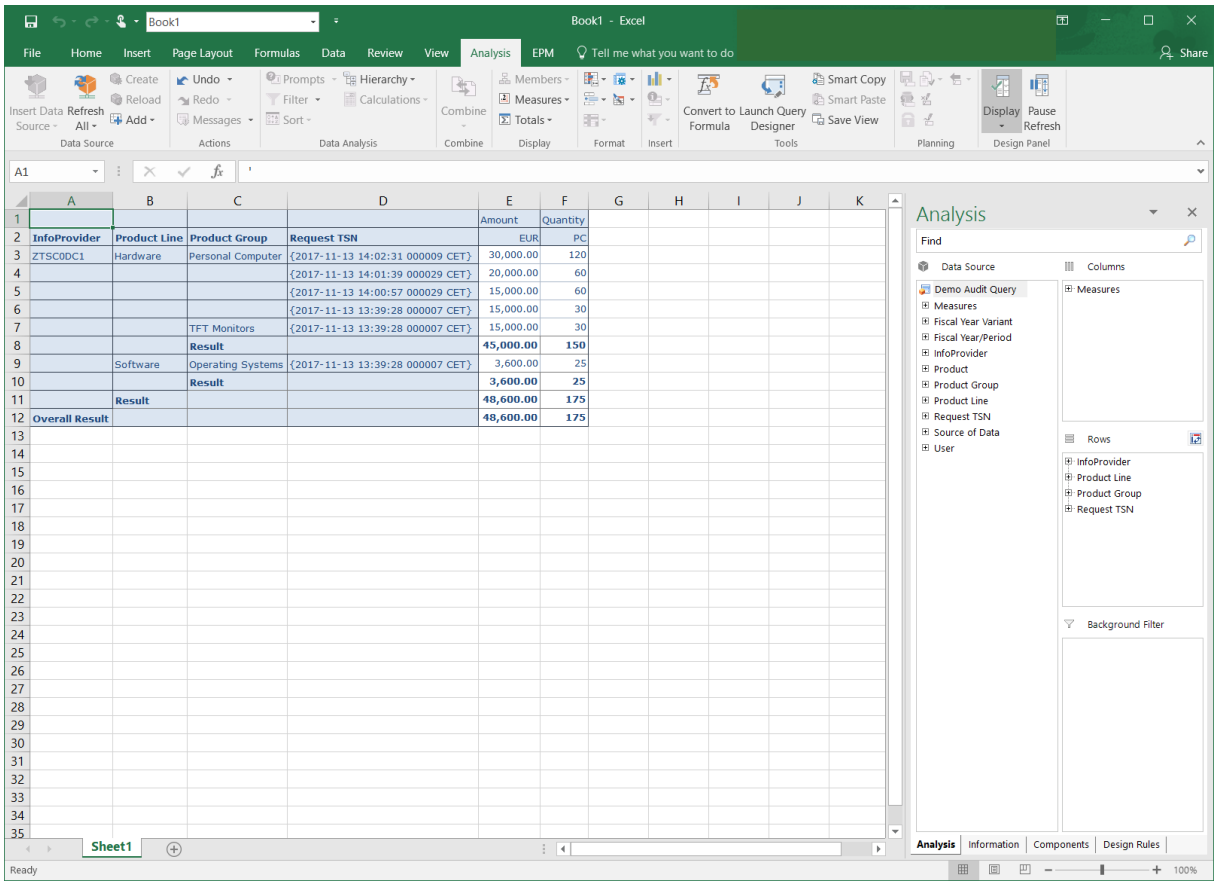
The following screen shows the definition of the audit query:



Executing the Audit Query ZTSC0H000_Q_AUDIT_00

When you execute the audit query, you see the following result: Using characteristic `0REQTSN`, you can track which values have been changed by the user of the input-ready query. You need to use property *Cumulate Values* for characteristic `0REQTSN` in the query definition.

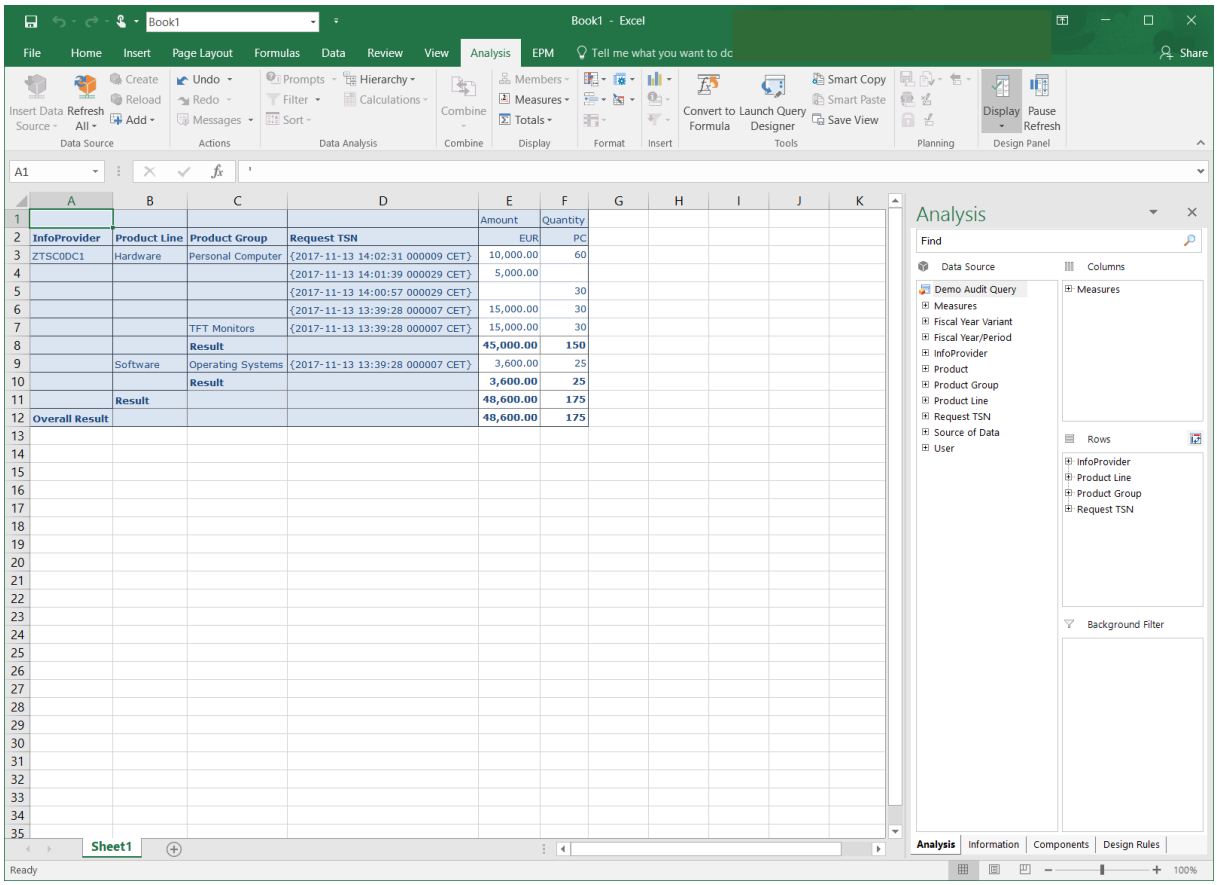
The following screen shows the executed query with the changes:



	A	B	C	D	E	F
1					Amount	Quantity
2	InfoProvider	Product Line	Product Group	Request TSN	EUR	PC
3	ZTSC0DC1	Hardware	Personal Computer	{2017-11-13 14:02:31 000009 CET}	30,000.00	120
4				{2017-11-13 14:01:39 000029 CET}	20,000.00	60
5				{2017-11-13 14:00:57 000029 CET}	15,000.00	60
6				{2017-11-13 13:39:28 000007 CET}	15,000.00	30
7			TFT Monitors	{2017-11-13 13:39:28 000007 CET}	15,000.00	30
8			Result		45,000.00	150
9		Software	Operating Systems	{2017-11-13 13:39:28 000007 CET}	3,600.00	25
10			Result		3,600.00	25
11		Result			48,600.00	175
12	Overall Result				48,600.00	175

If the audit query is executed without the property *Cumulate Values* for characteristic `0REQTSN`, you can track the changes (delta values).

The following screen shows the executed query with the delta values:



	A	B	C	D	E	F
1					Amount	Quantity
2	InfoProvider	Product Line	Product Group	Request TSN	EUR	PC
3	ZTSC0DC1	Hardware	Personal Computer	{2017-11-13 14:02:31 000009 CET}	10,000.00	60
4				{2017-11-13 14:01:39 000029 CET}	5,000.00	
5				{2017-11-13 14:00:57 000029 CET}		30
6				{2017-11-13 13:39:28 000007 CET}	15,000.00	30
7			TFT Monitors	{2017-11-13 13:39:28 000007 CET}	15,000.00	30
8			Result		45,000.00	150
9		Software	Operating Systems	{2017-11-13 13:39:28 000007 CET}	3,600.00	25
10			Result		3,600.00	25
11		Result			48,600.00	175
12	Overall Result				48,600.00	175

3.9.9 Names for CompositeProvider Fields in the Query

For every field of a CompositeProvider, unique names are used in the query. The InfoObject names are derived according to specific rules.

Names for CompositeProvider Fields in the Query

Derivation from...	InfoObject Name of CompositeProvider Field
Default The default is applied if there is no association to the field of the CompositeProvider, or if there is an association, but the *direct usage for reporting* flag is not set.	4<Name of CompositeProvider>-<Name of CompositeProvider field>
Association to InfoObject and *direct usage for reporting* flag is set	<name of associated InfoObject>

Derivation from...	InfoObject Name of CompositeProvider Field
Association to Open ODS View and *direct usage for reporting* flag is set	2F<name of associated ODS view>

3.9.10 Authorizations for CompositeProviders

To work with CompositeProviders, you need authorization object S_RS_HCPR. The analysis authorizations are not a combination of the authorizations of the participating InfoProviders.

To be able to use SAP HANA views, the SAP<SID> database user requires the authorization for DB schema _SYS_BI.

3.9.11 Transporting CompositeProviders

The CompositeProvider is integrated into the TLOGO framework of the SAP BW/4HANA system and can be transported.

The transport object is called HCPR.

During the export step, the system checks whether the individual parts of the CompositeProvider, the participating providers and InfoObjects, are transportable. It does not check whether all participating objects in the CompositeProvider are contained in the transport request. This means that you can transport a CompositeProvider without its providers.

For SAP HANA views, the system does not check whether they are in the target system. You need to transport these objects to the target system separately.

SAP HANA views contained in HDI containers managed in the XSA application are not transported. You have to transport these SAP HANA views manually. If you use ABAP-managed HDI containers, the SAP HANA transport container (HTC) is used to transport HDI objects.

In the target system, the CompositeProvider can then only be used for queries if all individual parts of the CompositeProvider are there.

3.9.12 Recommendations for Modeling CompositeProviders

Inner Join versus Left Outer Join

If possible, use an inner join. An inner join is always faster.

Referential Integrity

If referential integrity is guaranteed, you should set the corresponding flag on the *Output* tab. You then need to make sure that there is exactly one master data record for every value of the characteristic in the provider. If referential integrity exists, this can improve performance. You can also configure this setting later on.

> ⓘ Note
>
> If you have set the flag for referential integrity, you have to make sure in future that only one master data record exists for each characteristic value in the provider. If you do not do this, you might find that records are missing in the query result because, with referential integrity, fields are assigned with an inner join.

Using OLAP Functions in Mixed Scenarios with SAP HANAviews

If you make significant use of BW OLAP functions, we recommend only using BW-persisted InfoProviders. This ensures the best possible performance when using BW OLAP functions.

OLAP functions (such as exception aggregation) can only be executed optimized in SAP HANA if all fields used in the query have master data IDs (SIDs). SAP HANA views do not have SIDs

3.10 Document in the Document Store

In a suitable front end, you can document cells in a query with numbers. You can use these (text) documents on the backend store. TLOGO object. `Document Store` is used for this.

A document is identified by the name of the key figure and the filter for this key figure, which affects the cell in the query. The text length of a document is unlimited. You can assign a document to a hierarchy node or a subtotal. You can also assign a document to restricted and calculated key figures. To make it possible to add comments for calculated key figures, you have to assign a technical ID. Make sure that these are unique. As soon as the query contains characteristic **OREQTSN**, no comments can be created.

Documents can be reused in other queries if the filter selection is identical. A document is therefore visible in all queries that contain the same cell. You can even find documents from the drilldown in the selection and vice versa, as long as their common part is identical.

In the `Document Store`, you have to define which InfoProviders it will store. Queries that are based on these InfoProviders can work with comments from this document storage.

> ⓘ Note
>
> Note the authorization concept: In the query, you can document what you can see on the basis of the analysis authorizations for the query. You also require the authorizations for standard authorization object `S_RS_DOCA` (activities `READ` and `WRITE`) in order to read and edit documents.

Related Information

Document Support [page 443]
Creating a Document Store [page 314]

3.11 Query

A query is the formal definition of a multidimensional query. You use analytic queries to analyze the data in the SAP BW/4HANA.

The following sections provide details about the concepts of an analytic query:

Related Information

Characteristic [page 112]
Structure [page 112]
Structure Element [page 113]
Selection [page 114]
Formula [page 114]
Cell, Exception Cell and Help Cell [page 114]
Filter [page 115]
Condition [page 116]
Exception [page 117]
Reusable Query Components [page 117]
FEMS [page 141]
Transporting Queries [page 141]
These links take you to the documentation for query concepts.
Working with Queries [page 316]
This link takes to you to the documentation for modeling queries.

3.11.1 Characteristic

A characteristic is a classification group, such as company code, product, customer group, fiscal year, period, or region. It stipulates classification options for the dataset and can therefore be seen as a reference object for the key figures.

Special features of characteristics

- Time characteristics are characteristics with a time reference, such as date, month, calendar month, fiscal year/period.
- If a characteristic has attributes, texts or hierarchies, it is a master data-bearing characteristic.
  Master data is data that remains unchanged over a long period of time. Master data contains information that is always needed in the same way. References to this master data can be made in all InfoProviders.

  > ⓘ Note
  >
  > The company location is an example of an attribute for Customer. You use this to form customer groups for a specific region for example.

- If a characteristic has one or more hierarchies, it is a hierarchy basic characteristic.
  A characteristic hierarchy is the representation of a characteristic's values in a tree structure. Like master data, characteristic hierarchies can be used in all InfoProviders.

  > ⓘ Note
  >
  > A hierarchy for characteristic "Cost Center" makes it possible to consolidate cost centers into cost center groups for example.

Related Information

Including Characteristics [page 320]

3.11.2 Structure

A structure is an object that is created when a query is defined. It is freely definable. In a query, you can one structure, two structures or no structures at all.

A structure forms the basic framework of the axes in a table (rows or columns). You can combine structures freely with other characteristics on the axes.

A structure is made up of structural elements. The following types of structures exist:

- **Key figure structure**: The structural elements of a key figure structure are always based on key figure selections (basic key figures, restricted key figures and calculated key figures). You can use a maximum of one key figure structure per query.
- **Characteristic structure**: The structural elements of a characteristic structure cannot contain any key figure selections.

The setup of the structure determines the sequence and number of key figures or characteristic values in the columns and rows of the query. You can navigate through the structures in the executed query and set filters for them.

If the query contains two axes, a key figure structure in the columns and a characteristic structure in the rows for example, a table is created with fixed cell definitions.

> ⓘ Note
>
> You have to use two structures as a prerequisite for defining exception cells. You can override the value of cell values created implicitly at the intersection of two structural elements.

Related Information

Structure Element [page 113]
This link refers to a related concept.
Defining the Structure [page 327]
Defining Structure Elements [page 328]
These links refer to unrelated tasks.

3.11.3 Structure Element

A structure element is part of a structure.

Structure elements can be hierarchically arranged in a structure. They have common properties. There are two subtypes: Selections (with and without key figures) and formulas for creating calculated key figures.

Related Information

Selection [page 114]
Formula [page 114]
These links refer to related concepts.
Defining Selections [page 329]
Defining a Formula [page 344]
These links refer to related tasks.

3.11.4 Selection

A selection is a structural element that restricts a key figure or a characteristic to a certain selection of values.

Related Information

Defining Selections [page 329]
Selection Properties [page 330]

3.11.5 Formula

A formula is a structural element that uses a formula definition to recalculate the values of a key figure.

Related Information

Defining a Formula [page 344]
Formula Properties [page 346]

3.11.6 Cell, Exception Cell and Help Cell

The definition of a cell arises from the intersection of two structural elements from different structures. This implicit cell definition can be overwritten by an explicit definition. This defines an exception cell. You can add one or more help cells to an exception celll.

Cell

If you use two structures in a query, the system implicitly creates a generic cell definition at the intersection of two structural elements, thus defining the value to be presented in the cell.

Exception Cell

Cell-specific definitions allow you to define formulas and selection conditions for cells, thus overwriting the implicitly created cell values. You can create precisely one such explicitly defined exception cell for each cell. Exception cells allow you to design queries with a much higher degree of detail.

> ⚠ Caution
>
> Note that using exception cells can significantly increase the time it takes to execute your query.

You have the following options when defining an exception cell:

- As a reference cell that is used in an exception cell that is defined as a formula or selection,
- As a formula
- As a selection

The properties of an exception cell correspond to those of the corresponding structural element (formula or selection). To define how an exception cell is displayed, you can make settings that deviate from those in the column where the cell is situated.

Help Cell

You can also define one or more help cells for a cell. These do not have any direct connection with the corresponding structural elements. These cells are **not** displayed. They are used for help selections and for help formulas.

You have the following options when defining a help cell:

- As a formula
- As a selection

Related Information

Defining Exception Cells and Help Cells [page 358]

3.11.7 Filter

A filter is an object that describes a multidimensional extract of data from a data set.

You can use filters to restrict data to a certain business area, certain product groups, or certain time periods for example. By segmenting the data set, you can ensure that users or user groups only have access to the data that is relevant to them and can only see the relevant data areas in an application scenario.

The filter selection applies to the entire query. All the InfoProvider data is aggregated using the filter selection of the query. You can select characteristics from the InfoProvider's directory tree and restrict them to individual characteristic values. You can also select individual characteristic values or exactly one key figure.

Decide whether to base all the query data on the numerical values of just one key figure or to base it on multiple key figures. If you want to base the query on the numerical values of exactly one key figure, select the key figure in question as your filter. You can also use a restricted or calculated key figure for this.

By selecting default values, you can create a query that is based on extensive data, but for which only a specific part of the data is displayed when the query is executed. The data displayed here is the data that you are most interested in. You can then change the filter by adding new objects for example, and analyze the other data.

> **Note**
>
> Example: You want to analyze all customers in a query. In the query's start state however, you initally only want to see special customers.

Related Information

Defining Filters [page 362]

3.11.8 Condition

A condition is an object that restricts the view of the data in a query by hiding numbers that do not meet the defined condition(s).

By applying conditions to a query, you do not change any figures. You simply hide figures that are not relevant for you. Conditions therefore have no effect on the values displayed in the result rows. The results row of a query with an active condition corresponds to the results row of the query without this condition.

You can define multiple conditions for a query. Conditions are evaluated independently of each other. The result set is therefore independent of the evaluation sequence. The result is the intersection of the individual conditions. Multiple conditions are linked logically with AND. A characteristic value is only displayed if it meets all (active) conditions in the query.

Using conditions, you can restrict how you view query data as follows:

- **Threshold Values**: An entry is filtered independently of the other entries if its reference value has a specific relationship to a comparison value. An entry is **not** displayed for example if its reference value exceeds or is less than a specific threshold value.
- **Ranked List**: All entries for the displayed list or all entries for a logical section of the list (with multiple characteristics in the drilldown) are considered. Their relationship to each another determines whether the entry is displayed. For ranked lists, the sorting is switched on automatically when the condition is activated.

Related Information

Defining Conditions [page 367]

3.11.9 Exception

An exception is an object that allows you to highlight unusual deviations from the expected results. Deviations from defined threshold values are highlighted with a specified colour.

The definition of exceptions contains the following specifications:

- The specification of threshold values or intervals, which are given a priority (bad, critical, good). The priority given to an exception corresponds to preassigned colors that become more 'intense' the greater the deviation. Up to nine different shades of the traffic light colors (red, yellow, and green) are used.
- The specification of characteristic restrictions. Using characteristic restrictions, you specify which cells you want to apply the exception to.

Related Information

Defining Exceptions [page 370]

3.11.10 Reusable Query Components

A reusable query component is an object that is not created within the scope of a query definition, but either for an InfoProvider or an InfoObject.

You can use a reusable query component in all queries that are defined for this InfoProvider or with this InfoObject. When a reusable object is changed, the changes are applied everywhere that the object is used.

The following reusable objects can be defined for an InfoProvider:

- restricted key figure
- calculated key figure
- structure
- filter

The following reusable object can be defined for an InfoObject:

- variable

Related Information

Calculated Key Figure [page 118]
Restricted Key Figure [page 118]
Structure [page 119]
Filter [page 119]
Variable [page 119]
These links take you to the documentation for concepts.

Using Reusable Query Components [page 379]
Restricting Characteristics [page 363]
Defining Calculated Key Figures [page 379]
Defining Restricted Key Figures [page 382]
Defining the Structure [page 385]
Defining Filters [page 387]
Defining Variables [page 389]
These links take you to the documentation for tasks.

3.11.10.1 Restricted Key Figure

You can restrict a key figure of an InfoProvider by using one or more characteristic selections. To do so, you define a selection for a base key figure, a calculated key figure, or an already restricted key figure. It can then be reused as a restricted key figure in all queries for the underlying InfoProvider.

A restricted key figure is not an actual element in a query. It is a placeholder that references the restricted key figure created in the InfoProvider.

Related Information

Defining Restricted Key Figures [page 382]
Using Restricted Key Figures [page 384]
Overwriting Restricted Key Figures Locally [page 384]

3.11.10.2 Calculated Key Figure

You can recalculate the key figures of an InfoProvider using a formula. Calculated key figures consist of formula definitions that may contain basic key figures, restricted key figures or precalculated key figures. Calculated key figures are reusable.

A calculated key figure is not an actual element in a query. It is a placeholder that references the calculated key figure created in the InfoProvider.

Related Information

Defining Calculated Key Figures [page 379]
Using Calculated Key Figures [page 381]
Overwriting Calculated Key Figures Locally [page 381]

3.11.10.3 Structure

A structure created for an InfoProvider can be reused in all queries to this InfoProvider.

This kind of reusable structure is built in the same way as a structure that is created in a query (locally).

Related Information

Structure [page 112]
This link takes to you to the documentation for the concept of a (local) structure.
Using Structures [page 386]
Locally Overwriting a Structure [page 387]
These links take you to the documentation for the procedures for a reusable structure.

3.11.10.4 Filter

A filter created for an InfoProvider can be reused in all queries to this InfoProvider.

This kind of reusable global filter is structured in the same way as a filter that is created in a query (locally).

Related Information

Filter [page 115]
This link takes to you to the documentation for the concept of a (local) filter.
Reusing Filters [page 388]
Overwriting Filters Locally [page 389]
These links take you to the documentation for the procedures for a reusable filter.

3.11.10.5 Variable

A variable is a means of parameterizing a query or a query component, as a variable is filled with values only once the query is executed. You define a variable for an InfoObject. This variable is then available in all InfoProviders that use this InfoObject, and can be processed in many different ways.

Variables act as placeholders for characteristic values, hierarchies, hierarchy nodes, texts, or formula elements. The **variable type** indicates for which object you can use a variable.

Depending on the variable type, variables can be processed in various ways. The **processing type** of a variable determines how the variable is filled with a value at the runtime of the query.

Related Information

Overview of Variable Types and Processing Types [page 120]
Variable Type [page 121]
Processing Type [page 122]
Display Area [page 139]
These links take you to the documentation for query concepts.
Restricting Characteristics [page 363]
Restricting Characteristics by Characteristic Values [page 363]
Restricting Characteristics on Hierarchy Nodes [page 365]
These links take you to the documentation for query procedures.

3.11.10.5.1 Overview of Variable Types and Processing Types

When you create a variable, you may need to select a processing type - depending on the variable type. Depending on the variable type, only certain processing types are useful.

The following table gives an overview of the possible combinations of variable types and processing types.

Variable type	Processing type: *User Entry* (manual entry) and default value	Processing type: *Replacement Path*	Processing type: Exit	Processing type: *Authorization*
Characteristic Value: Characteristic Value Variable	Yes	Replacement from CURRENT MEMBER variable, query or variable	*Customer Exit*, *SAP Exit*, *SAP HANA Exit*	Yes
Hierarchy: Hierarchy Variable	Yes		*Customer Exit*, *SAP Exit*	
Hierarchy Nodes: Hierarchy Node Variable	Yes		*Customer Exit*, *SAP Exit*	Yes
Text: Text Variable	Yes	Replacement from InfoObject or variable	*Customer Exit*, *SAP Exit*	
Formula: Formula Variable	Yes	Replacement from InfoObject or variable	*Customer Exit*, *SAP Exit*, *SAP HANA Exit*	

Related Information

Variable Type [page 121]
Processing Type [page 122]

3.11.10.5.2 Variable Type

When creating a variable, the variable type indicates for which object you can use a variable.

The following table provides an overview of the variable types:

Variable Types

Type	Description
Characteristic Value: Characteristic Value Variable	Characteristic value variables represent characteristic values and can be used wherever characteristic values can be used. If you restrict characteristics to specific characteristic values, you can also use characteristic value variables instead of fixed values.
Hierarchy: Hierarchy Variable	Hierarchy variables represent hierarchies and can be used wherever hierarchies can be selected. If you restrict characteristics on hierarchies or presentation hierarchies, you can use variable hierarchies with hierarchy variables instead of fixed ones.
Hierarchy Nodes: Hierarchy Node Variable	Hierarchy node variables represent nodes in a hierarchy and can be used wherever hierarchy nodes can be used. If you restrict characteristics to hierarchy nodes, you can also use hierarchy node variables.
Text: Text Variable	Text variables represent a text and can be used in descriptions of queries, calculated key figures and structural components (selections and formulas). If the system finds either no values or multiple values for the reference characteristic when replacing text variables, and is thus unable to determine a unique value, it outputs the technical name of the text variables as the result.
Formula: Formula Variable	A formula variable represents numeric values and can be used in formulas and for selecting exceptions and conditions. **Example** By using a formula variable for the interest rate, you do not need to process the current interest rate until after you have executed the query.

Related Information

Restricting Characteristics [page 363]
Restricting Characteristics by Characteristic Values [page 363]
Restricting Characteristics on Hierarchy Nodes [page 365]

3.11.10.5.3 Processing Type

When you create the variable, you use the processing type to specify how the variable is filled with a value at query runtime.

The system supports the following processing types:

- *User Entry*: Manual Entry/Default Value
- *Replacement Path*: Replacement Path
- *Customer Exit*: Customer Exit
- *BRFplus Exit*: BRFplus Exit
- *SAP Exit*: SAP Exit
- *SAP HANA Exit*: SAP HANA exit
- *Authorization*: Authorization

Related Information

Manual Entry/Default Value [page 122]
Replacement Path [page 123]
Customer Exit [page 132]
BRFplus Exit [page 136]
SAP Exit [page 137]
SAP HANA Exit [page 137]
Authorization [page 139]

3.11.10.5.3.1 Manual Entry/Default Value

The processing type *User Entry* allows you to either enter the required value manually or use the default value for the variable when you execute the query.

The processing type *User Entry* is available for all variable types.

Manual Entry

If you select the processing type *User Entry* for an input-ready variable, you can manually enter the required value when executing the query.

Default Value

When you create the variable, you can specify a default value on the *Default Values* tab in the Variables Editor. If you enter a default value, this is shown in the input field of the variables screen for variables that are ready-for-input. For variables that are not ready-for-input, the default value is used as the variable value.

Related Information

Additional Tabs Available for Specific Processing Types [page 395]

3.11.10.5.3.2 Replacement Path

You use the *Replacement Path* processing type to specify the value that **automatically** replaces the variable when you execute the query.

The *Replacement Path* processing type can be used with characteristic value variables, text variables, and formula variables.

The following table provides an overview of the possibilities of the *Replacement Rule*:

Replacement Options Independent of Variable Type

Variable Type	Replacement with
Characteristic Value: Characteristic Value Variable	• CURRENT MEMBER variable • Query • Variable
Text: Text Variable	• InfoObject • Variable
Formula: Formula Variable	• InfoObject • Variable

Related Information

Replacement from Characteristic Value [page 124]
Replacement from CURRENT MEMBER Variable [page 130]
Replacement from Query [page 130]
Replacement from Variable [page 131]

3.11.10.5.3.2.1 Replacement from Characteristic Value

Text and formula variables with the processing type *Replacement Path* can be replaced with a corresponding characteristic value. The value of a formula variable has to be a number. The value of a text variable has to be a text.

Variables Wizard

On the *Variable - Replacement Rule* dialog in the second step of the variables wizard, you specify that the variable derives its value from an InfoObject.

Variables Editor

In the variables editor under *Replacement Rule*, the system indicates that a variable serves as the source for replacing the variable. This replacement rule **cannot** be modified.

You can still change the object that you want to use for the replacement however. Choose *Browse*. A dialog box appears where you can select a different object.

The following properties are still modifiable:

- *Replacment With:* Choose where the value of the variable will be determined from.
  - *Key* The value of the variable is replaced with the characteristic key.
  - *External Characteristic* (of the key) The value of the variable is replaced with an external value of the characteristic (external/internal conversion).
  - *Name* The value of the variable is replaced with the name of the characteristic.

    > ⚠ Caution
    >
    > Note that formula variables have to contain numbers in their names so that the formula variable represents a number after replacement.

  - *[1ATTRREF] Characteristic Reference (Constant 1)* - this setting is only available for formula variables.
  - *[0SIGNCH] Sign Change*: The variable value is replaced with a value of a hierarchy attribute. You need this setting for sign reversal with hierarchy nodes. This is only available for formula variables.

- *After Aggregation* (replacement of the variable after aggregation): This option is set by default. If you deselect this, the variable will be replaced before aggregation. It is not always possible to change this however. Replacement of the variable in the case of *[1ATTRREF] Characteristic Reference (Constant 1)* always takes place **before** aggregation. In the case of *Name* and *[0SIGNCH] Sign Change*, it takes place **after** aggregation.
- *Interpret Result As Boolean Value (True=1; False=0)* This function is only available for formula variables. If you select this setting, the formula variable will **not** be replaced with a value (the key value for example) but merely specifies the existence of a value.
  - The value of the variable is 0 when the value to be replaced is a string with only blank spaces or a zero.
  - The value of the variable is 1 when the value to be replaced is a string with characters that are not blank spaces or a number does not equal 0.

    > ✤ Example
    >
    > You want to calculate paid amounts only.
    >
    > Create a formula variable with processing type Replacement Path and replacement rule InfoObject in order to replace the variable with the value of an attribute. Choose the attribute **Paid**. (The master data attribute **Paid** shows open invoices with a blank space " " and paid invoices with the value "X".) Select *Interpret Result As Boolean Value (True=1; False=0)*.
    >
    > You then get a 0 for blanks and a 1 for the value "X". By a multiplication with this kind of formula variable, all unpaid amounts in the query are then omitted. You have met your objective of only having paid amounts.

Under *Details*, you can make the following settings:

- *If Interval Use*:
  - *From Value*
  - *To Value*
  - *Difference*

  If the replacement value found is an interval, you can use this setting to specify whether the lower interval limit (from value) or the upper interval limit (to value) is used as the replacement value.
  For formula variables, you can also choose the interval difference as the replacement value. Using this setting, you can easily calculate the difference of period boundaries. The interval difference is obtained by subtracting the From value from the To value and then adding 1.

  > ✤ Example
  >
  > The period January to March therefore has a 3 (To value 3 minus From value 1 plus1).

- Offset Settings:
  - *Offset Start*
  - *Offset Length*:

  You use this function to specify the start and length of the text output.

  > ✤ Example
  >
  > You want to create a text variable for the fiscal year.
  >
  > Create a text variable with processing type Replacement Path and replacement rule InfoObject in order to replace the variable with the value of an attribute. Under *Replacement with*, choose replacement by key and define the offset start at value 2 and the offset length as 2.

The fiscal year contains four digits. The output of the value begins after the second digit and shows two digits. In the query, the text variable for fiscal year 2005 is therefore replaced with the value "05".

Related Information

Calculating with Formula Variables [page 126]
Offset for Text Variables [page 128]
Additional Tabs Available for Specific Processing Types [page 395]

3.11.10.5.3.2.1.1 Calculating with Formula Variables

The following example shows how you can calculate with formula variables using processing type *Replacement Path* with replacement from a characteristic value with the properties/values of the characteristic. The formula is replaced here in accordance with the cell context.

Using Formula Variables in Calculated Key Figures and Formulas

The replacement of formula variables with the processing type *Replacement Path* from a characteristic value acts differently in calculated key figures and formulas:

- If you use a formula variable with "Replacement from the Value of an Attribute" in a calculated key figure, the system automatically adds the drilldown according to the reference characteristic for the attribute. The system then evaluates the variables for each characteristic value for the reference characteristic. The calculated key figure is then calculated and, subsequently, all of the other operations are executed (all additional, calculated key figures, aggregations, and formulas). **Before** aggregation with the reference characteristic, the system therefore only calculates the operators which are assembled in the calculated key figure itself.

  ⚠ Caution

  Note that calculation **before** aggregation can have a negative impact on performance.

- If you are using a formula variable with Replacement from the Value of an Attribute in a formula element, the variable is only calculated if the reference characteristic is uniquely specified in the corresponding row, column, or in the filter.

✤ Example

You might want to multiply basic key figure **Sales Quantity** with the current value for attribute **Price** for characteristic **Material** for example. To do this, you have to use a formula variable that has been defined accordingly. Processing type *Replacement Path*, replacement from characteristic **Material**, Replace Variable with an Attribute Value, here the attribute **Price**.

Replacement from the Value of an Attribute

When using formula variables, you can create a reference to the reference characteristic of the variable by taking the value of the variables from attribute *[1ATTRREF] Characteristic Reference (Constant 1)*. This object is a dummy attribute that is available with each characteristic. It is used to create a reference to the characteristic, which is **not** used for aggregation. By choosing this attribute, you can influence the aggregation behavior of calculated key figures in a targeted way and can improve performance during calculation.

Example

Master data table for material

Material	Price
Material 1	10
Material 2	20

InfoProviders

Material	Month	Sales Quantity
Material 1	1.2002	5
Material 1	2.2002	6
Material 2	1.2002	7
Material 2	2.2002	8

The example query has the following objects:

- Material Price (V1) is a variable that is replaced by the material price.
- K1 = Material Price (V1) is a calculated key figure.
- F1 = Material Price (V1) is a formula.
- K2 = Material Price (V1) * Sales Quantity is a calculated key figure.
- F2 = Material Price (V1) * Sales Quantity is a formula.
- K3 = K1 * Sales Quantity is a calculated key figure in which calculated key figure K1 is further processed.

Query View 1

Material	Sales Quantity	K1	F1	K2	F2	K3
Material 1	11	10	10	110	110	110
Material 2	15	20	20	300	300	300
Result	26	30	X	410	X	780

Query View 2

Month	Sales Quantity	K1	F1	K2	F2	K3
1.2002	12	30	X	190	X	360
1.2002	14	30	X	220	X	420
Result	26	60	X	410	X	780

In the results rows, columns F1 and F2 are not specified, because the material value is not unique. The system cannot calculate F1 and F2 in the overall Query View 2 either. K1 only aggregates the prices. K3 multiplies these aggregated prices with the aggregated sales quantities. The results from K1 and K3 usually have no meaning. Calculated key figure K2 delivers the required results.

3.11.10.5.3.2.1.2 Offset for Text Variables

The following example shows the settings that can be made for text variables using processing type *Replacement Path* with replacement from a characteristic in terms of the properties *Replacement With* (key or external attribute of the key) together with *Offset Start* and *Offset Length*.

The following table shows the internal and external display of the key for the characteristic values fiscal year variant K4 and fiscal year/period 2002/008 without offset settings:

Internal and External Display of the Key without Offset Settings

Replacement With	**Fiscal Year 0FISCYEAR**	**Fiscal Year / Period 0FISCPER**
Internal display (key)	K42002	K42002008
External display (external characteristic value of the key)	K4/2002	K4/008.2002

Internal Display (Key)

Fiscal Year 0FISCYEAR

- If you want to replace K42002 with 2002 in the query, you have to set the offset start to 2 and the offset length to 4 for the characteristic fiscal year 0FISCYEAR.

Fiscal Year/Period 0FISCPER

- If you only want to display the year in the text output, set the offset start to 2 and the offset length to 4. In the query, K42002008 is replaced by 2002.
- If you want to display the year and period in the text output, set the offset start to 2 and the offset length to 7. In the query, K42002008 is replaced by 2002008.

The following table illustrates how you can influence the internal display of the key for the characteristic values fiscal year variant K4 and fiscal year/period 2002/008 using offset settings:

Internal Display of the Key with Offset Settings

Characteristic	Offset Start	Offset Length 4
0FISCYEAR	Offset Start 2	Offset Length 4
0FISCPER	Offset Start 2	• Offset length 4 for year • Offset length 7 for year and period

External Display (External Characteristic Value of the Key)

Fiscal Year 0FISCYEAR

- If you want to replace K4/2002 with 2002 in the query, you have to set the offset start to 3 and the offset length to 4.

Fiscal Year / Period 0FISCPER

- If you want to display only the year in the text output, set the offset start to 7 and the offset length to 4. In the query, K4/008.2002 is replaced by 2002.
- If you only want to display the period in the text output, set the offset start to 3 and the offset length to 3. In the query, K4/008.2002 is replaced by 008.
- If you want to display the year and period in the text output, set the offset start to 3 and the offset length to 8. In the query, K4/008.2002 is replaced by 008.2002.

The following table illustrates how you can influence the external display of the key for the characteristic values fiscal year variant K4 and fiscal year/period 2002/008 using offset settings:

External Display of the Key with Offset Settings

Characteristic	Offset Start	Offset Length 4
0FISCYEAR	Offset Start 3	Offset Length 4
0FISCPER	• Offset length 7 for year • Offset start 3 for period • Offset start 3 for year and period	• Offset length 4 for year • Offset length 3 for period • Offset length 8 for year and period

3.11.10.5.3.2.2 Replacement from CURRENT MEMBER Variable

The variable derives its value from a CURRENT MEMBER variable.

To define filters that relate to the values of a time characteristic in the drilldown of a query during analyses, define a restricted key figure using a CURRENT MEMBER variable. This is a variable with the replacement path CURRENT MEMBER.

The CURRENT MEMBER variable can be used in restricted key figures as either the lower limit or the upper limit in combination with an offset. Only one restriction with a CURRENT MEMBER variable is allowed per restricted key figure however.

Related Information

CURRENT MEMBER Variables [page 607]

3.11.10.5.3.2.3 Replacement from Query

Characteristic value variables with the Replacement Path processing type can be replaced with the results of a query. The variable is replaced from the data result of the selected query.

Variables Wizard

On the *Variable - Replacement Rule* dialog in the second step of the variables wizard, you specify that the variable derives its value from the data result of a query. You then select the required query with *Browse*.

> ⚠ Caution
>
> Note that, for technical reasons, variables with processing type Replacement from Query **cannot** be used in restricted key figures or in selection structure elements.

Variables Editor

On the *Replacement Path* tab of the variables editor, you can select another query if you want to use its data result in future. In addition, you can specify whether this view should be used at the start of the query.

Related Information

Additional Tabs Available for Specific Processing Types [page 395]
Example: Using a Variable with Replacement from Query [page 391]

3.11.10.5.3.2.4 Replacement from Variable

Characteristic value variables, hierarchy variables, text variables and formula variables with processing type Replacement Path can be replaced by the value from a different variable, the source variable. The value is determined either from the key, the external attribute of the key, the description, or the attribute value. You can also define an offset start and offset length. The variable is replaced in the Variables screen with each data release.

The target variable and source variable must meet the following prerequisites:

Prerequisites for Replacement by Variable

Object	Properties
Target variable: The variable to be replaced should	• **not** be input-enabled • Screen area *Single Value*
Source variable: The variable that is used to make the replacement	• Not a hierarchy node variable • Input ready • Must exist in the query • Screen area *Single Value* or *Interval*

Variables Wizard

On the *Variable - Replacement Rule* dialog in the second step of the variables wizard, you specify that the variable derives its value from a variable. You then select the required variable by choosing *Browse*.

Variables Editor

In the variables editor under *Replacement Rule*, the system indicates that a variable serves as the source for replacing the variable. This replacement rule **cannot** be modified.

You can still change the object that you want to use for the replacement however. Choose *Browse*. A dialog box appears where you can select a different object.

The following properties are still modifiable:

- *Replacment With:* Choose where the value of the variable will be determined from.
  - *Key* The value of the variable is replaced with the characteristic key.

- *External Characteristic* (of the key) The value of the variable is replaced with an external value of the characteristic (external/internal conversion).
- *Name* The value of the variable is replaced with the name of the characteristic.
- *[1ATTRREF] Characteristic Reference (Constant 1)* (attribute value of the reference characteristic): The value of the variable is replaced with the value of an attribute of the characteristic that the source variable is defined on. This setting is only available for formula variables.

  > ⓘ Note
  >
  > If the attribute is **time-dependent**, the system **cannot** determine the variable value or does **not**replace the variable with the expected value, since the key date of the query is not properly taken into account if it is variable.
  >
  > If the attribute and the target characteristic have the same basic characteristic, and this is compounded, the target variable must contain the uncompounded value. Proceed as follows:
  >
  > - The source characteristic is either the characteristic that the source variable belongs to or the attribute that the replacement will be made from. If the basic characteristic of the source characteristic and the basic characteristic of the target variable are the same, and the internal value is to be transferred, the system directly applies the uncompounded value.
  > - In all other cases (if the basic characteristics are not the same, or replacement is to be made from the external format or the text), you must cut the uncompounded value with the offset and length specifications out of the target variable definition.

- *[0SIGNCH] Sign Change*: The value of the variable is replaced with a value of a hierarchy attribute. You need this setting for sign reversal with hierarchy nodes. This is only available for formula variables.

- *After Aggregation* (replacement of the variable after aggregation): This option is set by default. If you deselect this, the variable will be replaced before aggregation. It is not always possible to change this however. Replacement of the variable in the case of *[1ATTRREF] Characteristic Reference (Constant 1)* always takes place **before** aggregation. In the case of *Name* and *[0SIGNCH] Sign Change*, it takes place **after** aggregation.
- *Interpret Result As Boolean Value (True=1; False=0)* As in the case of replacement from a characteristic value, this function is only available for formula variables. If you select this setting, the formula variable will **not** be replaced with a value (the key value for example) but merely specifies the existence of a value.

Under *Details* - as in the case of replacement from a characteristic value - you can define the values for using an interval (*If Interval Use*), as well as the *Offset Start* and *Offset Length* for the text output.

Related Information

Replacement from Characteristic Value [page 124]
Additional Tabs Available for Specific Processing Types [page 395]

3.11.10.5.3.3 Customer Exit

The processing type *Customer Exit* allows you to set up a processing type for variables that is specially tailored for your requirements. A customer exit is designed as an enhancement that you can configure with customer-

specific logic. The customer exit enables you to generate default values for variables ,or create variable values from specified variables automatically. You can also check all the variables after they have been entered.

You can process all variable types (characteristic value variables, hierarchy node variables, hierarchy variables, formula variables, and text variables) using the processing type *Customer Exit*.

> ⚠ Restriction
>
> Please note that processing type *Customer Exit* **cannot** be used in conjunction with external SAP HANA views. Processing type *SAP HANA Exit* is provided for this purpose.

The prerequisite here is that you need to create the enhancement implementation for the enhancement spot `RSROA_VARIABLES_EXIT_BADI` (located in package `RSROA_VAR`) in the ABAP Development Tools. The BAdI implementation includes the interface `IF_BADI_INTERFACE`.

> ✤ Example
>
> You want to derive a characteristic value from another one. The InfoProvider contains the calendar day. You also want to see the cumulative value for the associated period in the query (starting with the first day of the month). Define a variable for the first day of the month with processing type *Customer Exit*.
>
> If you now enter the current calendar day (for example, 6/19/2015), a start date of 6/1/2015 appears in the customer exit, and the cumulated value of this period can be displayed.

Article	Sales 6/19/2015	Sales 6/1/2015 - 6/19/2015
A	5	263
B	7	157
C	2	67

Related Information

Dependencies in Customer Exit Variables [page 133]

3.11.10.5.3.3.1 Dependencies in Customer Exit Variables

If you execute a query that contains variables of processing type *Customer Exit*, there are three possible times when the exit can be called: Before the variable is entered (default values can be generated), after the variable is entered (variable values can be derived from other entries) and after variable processing is complete (to check the variable values).

Defining when the enhancement is called

This enhancement spot `RSROA_VARIABLES_EXIT_BADI` is called **multiple times** when reports are executed. Parameter I_STEP dictates when the enhancement is called.

The following table provides an overview of the values of parameter I_STEP:

Value for Parameter I_STEP

Value for Parameter I_STEP	Description
I_STEP = 1	Call directly before the variable is entered.
I_STEP = 2	Call directly after the variable is entered. This step is only used executed for non-input-ready variables, which could **not** be filled in I_STEP = 1. You can find an exception to this rule in parameter C_CHECK_AGAIN.
I_STEP = 3	Use this call to check the values of the variables. If an exception (RAISE) is triggered, the Variables screen appears again. I_STEP = 2 is then executed again.
I_STEP = 0	The enhancement is **not** called from the variables screen. The call can originate from the authorization check or from the monitor.

If a query contains multiple variables with customer exits, then the order of the variables is not defined in one step.

> ⓘ Note
>
> If variable 2 needs to be derived from specifiable variable 1 and variable 3 needs to be derived from variable 2, then the value for variable 2 must be generated in STEP 2 when variable 3 is derived. You can store the value for variable 2 in the buffer, in order to avoid the need to calculate it several times.

Method Signature

All parameters with names beginning with `I_` are importing parameters. All parameters with names beginning with `C_` need to be filled in the method.

Parameters of Method "Process"

Parameter	Description
I_VNAM	Technical name of variable to be edited.
I_VARTYP	Variable type of variable to be edited
I_IOBJNM	Characteristic name of variable to be edited

Parameter	Description
I_S_COB_PRO	Characteristic properties
I_S_RKB1D	Properties of the query, in particular the fields COMPUID (query name) and INFOCUBE (InfoProvider)
I_PERIV	Fiscal year variant (if specified and if unique)
I_T_VAR_RANGE	Values already known from other variable; see table below *Fields of Structure RRRANGEEXIT*.
I_STEP	See table above *Values for Parameter I_STEP*
C_T_RANGE	Return values
C_NO_SCREEN	If value ''X' is set in STEP 1, the system does not display the variable on the variable screen.
C_CHECK_AGAIN	If value ''X' is set in STEP 1, the system also processes the variable in STEP 2.
CX_RS_ERROR	Exception: An error occurred during processing.

Specifying values of other variables

When the enhancement spot `RSROA_VARIABLES_EXIT_BADI` is called, the system passes the currently available values of the other variable in table `I_T_VAR_RANGE`. The table type is `RRS0_T_VAR_RANGE`, and row type `RRS0_S_VAR_RANGE`. references structure `RRRANGEEXIT`.

The following table provides an overview of the values of the fields in structure `RRRANGEEXIT`:

Field in Structure RRRANGEEXIT

Field	Description
VNAM	Name of the variable
IOBJNM	Name of the InfoObject
SIGN	(I)ncluding [ ] or (E)xcluding] [
OPT	Operator: EQ =, BT →→, LE →→, LT →, GE →→, GT →, CP, and so on
LOW	Characteristic value
HIGH	Characteristic value of upper limit for intervals or node InfoObject for hierarchy nodes

In the following example, the current month is taken from an input-enabled variable MONTH. This is then used to generate an interval that includes all months from January ('01') up to the current month. The customer exit variable CUMMONTH contains the interval as the value.

Sample Code

```
DATA: L_S_RANGE TYPE RSR_S_RANGESID.
DATA: L_S_VAR_RANGE TYPE RRRANGEEXIT.
CASE I_VNAM.
WHEN 'CUMMONTH'.
IF I_STEP = 2. "after the popup
READ TABLE I_T_VAR_RANGE INTO L_S_VAR_RANGE WITH KEY VNAM = 'MONTH'.
IF SY-SUBRC = 0.
CLEAR L_S_RANGE.
L_S_RANGE-LOW = LOC_VAR_RANGE-LOW(4)."low value, z. B. 200601
L_S_RANGE-LOW+4(2) = '01'.
L_S_RANGE-HIGH = LOC_VAR_RANGE-LOW. "high value = input
L_S_RANGE-SIGN = 'I'.
L_S_RANGE-OPT = 'BT'.
APPEND L_S_RANGE TO E_T_RANGE.
ENDIF.
ENDIF.
ENDCASE.
```

3.11.10.5.3.4 BRFplus Exit

Variables of processing type *BRFplus Exit* make it possible to access the functions of the SAP rule engine *Business Rules Framework plus* (BRFplus) in SAP BW/4HANA.

In the *BW Modeling Tools*, you assign a a BRFplus function to a variable of this processing type. These BRFplus functions must have a corresponding BRFplus context (*signature*) and a corresponding result data object (*result object*).

A BRFplus function is suitable if it has one of the following result data objects:

Variable Type	Result Data Object	Data Object Type
Text Variable	E_RS_VARIABLE_TEXT_RESULT	Element Text
Formula Variable	E_RS_VARIABLE_FORMULA_RESULT	Element Number
Other Variable Type	E_T_RS_VARIABLE_RES_RANGE	Table with DDIC binding for table types RSSB_T_RANGE

In the context signature, the following data objects can also be used:

Data Object Name	Type
I_VNAM	Element (text)
I_VARTYP	Element (text)
I_IOBJNM	Element (text)
I_COMPID	Element (text)

Data Object Name	Type
I_INFOPROV	Element (text)
I_PERIV	Structure with DDIC binding for structure RRKB1F
I_T_VAR_RANGE	Table with structure with DDIC binding for structure RRRANGEEXIT
I_STEP	Element (number)
E_NO_SCREEN	Element (boolean)
E_CHECK_AGAIN	Element (boolean)

You can define the BRFplus function from any of the following modes: *Function Mode*, *Event Mode* or *Function and Event Mode*. If required, you can also use rule sets to define the logic.

Related Information

Using a Variable with Replacement from BRFplus [page 392]

3.11.10.5.3.5 SAP Exit

With the *SAP Exit* processing type you can use variables delivered by SAP within BI Content.

Please note that you can use the variables provided by SAP only once they exist as an active version. You may have to activate these delivered variables before you can use them.

You can find the variables provided by SAP by searching for object type ELEM with subtype VAR. The technical names of delivered SAP objects always begin with a number.

> ### Example
>
> If you want to define a query that only ever displays the data from the current month, drag the delivered variable **current month** (technical name `0CMONTH`) for the characteristic **calendar year/month** (technical name `0CALMONTH`) into the query filter.

> ### Restriction
>
> Please note that the *SAP Exit* processing type is **not** possible with external SAP HANA views. The *SAP HANA Exit* processing type is available for this.

3.11.10.5.3.6 SAP HANA Exit

Processing type *SAP HANA Exit* is a specific form of processing type *Customer Exit*. It allows you to use an AMDP class implementation to determine default values for variables. We particularly recommend using

the *SAP HANA Exit* in queries that you want to generate an external SAP HANA view for, and which special requirements apply for in terms of determining the variable values. In the query itself, the use of this variable type has no advantages. We recommend using SAP exit or customer exit variables.

Definition of Variables

In the *BW Modeling Tools*, you create variables of processing type *SAP HANA Exit*.

You can process variables of types characteristic value variables and formula variables with processing type *SAP HANA Exit*. If you use these for a formula variable, the entry type must **not** be **optional**.

A variable with processing type *SAP HANA Exit* represents a *single value* or multiple single values. In the case of multiple single values, the variable **cannot** be ready for input. It is therefore **not** possible to map characteristic values containing characters ' or ,.

The moment when the enhancement implementation is called depends on whether the variable is input-enabled:

- If the variable is input-enabled, the SAP HANA exit procedure is called exactly once **before** the variable screen. The variable is then automatically given the default values of the other variables in the procedure.
- If the variable is **not** input-enabled, the SAP HANA exit procedure is called exactly once **after** the variable screen. The variable is then given the values from the variable screen in the procedure.

Enhancement Implementation

In the ABAP Development Tools, you create the enhancement implementation for the enhancement spot `RSROA_VARIABLES_HANA_EXIT`. This is contained in package `RSROA_VAR`. The BAdI implementation class requires an AMDP class containing the following interfaces: `IF_BADI_INTERFACE`, `IF_AMDP_MARKER_HDB` and `IF_RSROA_VAR_HANA_EXIT`.

Example

As an example, you can use class CL_RSROA_HANA_EXIT_FALLBACK. Implement the following methods. After the method name, you have to enter the addition `by database procedure for hdb language SQLSCRIPT`. The code must comply with SAP HANA SQLSCRIPT syntax.

1. `if_rsroa_var_hana_exit~ get_properties`
   - Parameter `C_IS_ACTIVE` should have the value 'X'. This stipulates that this coding should be used to define the value of the variable
     The system can contain no more than **one** BAdI implementation with the value 'X for the query/ variable name combination.
   - Parameter `C_TS_VNAM_INDEX` defines the allocation of variable values from the query to the `I_VAR_VALUE_X` parameters of method `if_rsroa_var_hana_exit~process`. The specification of row `{VNAM_COUNTRY, 1}` in parameter `C_TS_VNAM_INDEX` stipulates that the value of variable `VNAM_COUNTRY` is assigned to parameter `I_VAR_VALUE_1`.
     A maximum of 20 entries of this type can be made.
2. `if_rsroa_var_hana_exit~ process`

- Parameter `c_value` should specify the value of the variable. If the variable represents multiple single values, they must be in a comma-separated format, for example 'DE' ,'AT'.

3.11.10.5.3.7 Authorization

With the *Authorization* processing type you can specify that a variable is **automatically** filled with the values from the user authorization. When the user opens a query, the system automatically selects the data according to his or her authorizations.

The *Authorization* processing type can be used with characteristic value variables and hierarchy node variables. As a precondition, you must have maintained the authorizations.

3.11.10.5.4 Display Area

When you create a variable, you choose the display area for the *Var. Represents* field.

The table below provides an overview of the possible display areas:

Display Areas for Variables

Display Area	Description
Single Value: single value	The variable represents exactly one value.
Interval: interval	The variable represents a specific 'from' value and a specific 'to' value, i.e. an interval.
Multiple Single Values: multiple single values	The variable represents multiple single values. This setting is useful in hierarchy nodes, for example to allow you to enter multiple single nodes.

Display Area	Description
Selection Option: selection option	The variable represents any combination of single values and intervals. In addition, when you select values for variables, you can use operators (>,<,=, and so on), use the contains pattern option (for example, search for all values that begin with A: A*), and exclude certain values (by specifying the values for which you do not want to search). **⚠ Caution** Note that you can only use the selection option variables with the **Contains Pattern** option in the filter. You cannot use them in the dynamic filter. The **Contains Pattern** option is also only supported for characteristics of type CHAR. The option is not supported for numeric characteristics.
Vorberechnet: Precalculated Value Set	The variable represents a precalculated value set of characteristic values. You can use this variable type to select value sets created for an InfoObject at query runtime.

3.11.11 Linked Component

A linked component can be automatically synchronized whenever changes are made to the corresponding source component.

Example scenario: You have two highly similar InfoProviders, IP_A and IP_B. You have created the query Q_A for IP_A. You now want to create the query Q_B for InfoProvider IP_B. You want this query to be very similar to query Q_A and to be automatically adjusted whenever changes are made to query Q_A.

To do this, you use the link component concept: You create the linked target query Q_B for source query Q_A. This is more than just a copy, as the system also retains the mapping information. This mapping information makes it possible to synchronize the queries.

ⓘ Note

For queries, the system supports mapping of the following query components: InfoObjects, hierarchies, restricted and calculated key figures, and variables.

Note that certain query settings cannot be mapped. Filter entries or currency translations/unit conversions **cannot** be adjusted.

If this makes it impossible to synchronize the query, you can create it as a standalone copy and make the adjustments there.

Related Information

Creating a Link Component [page 445]
Apply Changes from the Source Component [page 446]
Changing a Target Component [page 447]

3.11.12 FEMS

FEMS (**F**orm **E**le**M**ent **S**election) refers to the effectively acting local filter.

To calculate the result of a query, the elementary building blocks required for query generation are defined. A building block of this kind is generally a key figure with a filter. This is created either directly from a selection or indirectly as the operand of a calculated key figure. The filter is made up of all participating filters.

> ### Example
>
> If you have created a calculated key figure **Sales in North America** for example, with the formula **Sales in USA + Sales in Canada**, and use this calculated key figure in a selection (structural element) containing a filter on the year 2015, two building blocks are required for the calculation of the result:
>
> 1. Sales in USA, 2015
> 2. Sales in Canada, 2015

This kind of effective local filter is known as a FEMS.

If multiple structures or nestings of calculated and restricted key figures are used, all of them go into the FEMS in the filter acting on the operand.

Related Information

Blog about FEMS when using a SAP HANA database

3.11.13 Transporting Queries

The query is integrated into BW's TLOGO framework and can be transported.

The transport object is called ELEM.

All components of the query are written to the transport request individually. The number of entries in the transport request therefore depends on the complexity of the query. All required generation steps are performed on the target system.

You can define transport requests for query components package by package. In addition to this, it is possible to use a standard request for all development packages that are **not** covered by package-specific transport requests. The contain query components only, such as queries, filters, restricted key figures, calculated key figures and variable definitions.

Once the various query components have been saved, they are written to the relevant (package-specific or general) transport request. You do not need to select a transport request.

3.12 Source System

You use source systems in SAP BW/4HANA to set up the connection to a source for the data transfer.

Source systems are displayed in the BW Modeling tools in the *Project Explorer* view in the *Data Sources* tree. You can create source systems here and call source system functions.

The following types of source system are available:

- ODP source systems, to provide data via the ODP data replication API
- SAP HANA source systems
- Big Data source systems
- Flat files that metadata is maintained for manually and is provided using a file interface

Related Information

Working with Source Systems [page 455]

3.12.1 Transferring Data with an SAP HANA Source System

The SAP HANA source system provides central and unified access to sources that are provided using SAP HANA smart data integration or SAP HANA smart data access. It also rovides access to tables, views or native DataStore objects from the local SAP HANA databse or a SAP HANA tenant databse (Multitenant Database Container).

The SAP HANA source system allows various source system configurations that make it possible to access tables, views or native DataStore objects from database management systems:

- SAP HANA Smart Data Access
  For databases that are connected with SAP BW/4HANA by SAP HANA Smart Data Access. When defining the source system, you define which remote source (database) and which schema to access.
  For databases and other external systems that are connected with SAP BW/4HANA by SAP HANA Smart Data Integration.
- Local SAP HANA Database Schema
  For database tables, views, or native DataStore objects of the SAP HANA database from SAP BW/4HANA.
- SAP HANA Tenant Database Schema
  For database tables, views or native DataStore objects of a database container of the SAP HANA database of the SAP BW/4HANA system.

Use

You can access the data virtually using Open ODS views and CompositeProviders. Data transfer is supported in full mode for all sources, provided the source supports this (generic delta).

For many data provisioning adapters, real-time replication is supported with the SAP HANA source system.

Related Information

Creating a Source System of Type SAP HANA Local Database Schema [page 460]
Creating a Source System of Type SAP HANA Smart Data Access [page 461]
Creating a Source System of Type SAP HANA Tenant Database Schema [page 462]
Creating an SAP HANA DataSource [page 475]

3.12.1.1 Connection Types for the SAP HANA Source System

Various connection types for the SAP HANA source system are available: Connection types that you can use to access source objects in a schema of the local SAP HANA database or in a schema of a SAP HANA tenant database, and a further connection type that you can use to access objects belonging to a source connected with SAP HANA Smart Data Access or SAP HANA Smart Data Integration.

Local SAP HANA Database Schema

You use this option if a particular schema in the local SAP HANA database is to be used as the source system. You can then access tables, views or native DataStore objects belonging to the SAP HANA schema defined in the source system configuration.

You specify the schema as a connection parameter.

> → Tip
>
> You can also create a source system of this connection type directly when creating an Open ODS view.

SAP HANA Smart Data Access

Choose this option if you want the source system to be an external data source that is connected by Smart Data Access with the SAP HANA database on which the SAP BW/4HANA system is running. With this connection type, you can also use sources provided by SAP HANA Smart Data Integration. The connection to the external sources takes place using data provisioning adapters.

As connection parameters, you specify the remote source that connects the source to SAP HANA as well as an area of the source for which access is available via the source system. Depending on the remote source, and therefore, on the adapter, the area is specified either by specifying the remote database and an owner or schema, or, if the source has a hierarchical folder structure instead of schemas, using a path prefix.

SAP HANA Smart Data Integration

SAP HANA Smart Data Integration allows you to access data from various sources in real time or to transfer the data to the SAP BW/4HANA. It enables real-time high-speed data acquisition and transferal of masse data and federations. The connection to the external sources takes place using data provisioning adapters. Smart data integration provides adapters for a range of sources and allows custom adapters to be developed using an SDK.

> → Tip
>
> You can also create a source system of this connection type directly when creating an Open ODS view.

SAP HANA Tenant Database Schema

SAP HANA supports multiple isolated databases in a single SAP HANA system. These are known as tenant databases. A SAP HANA system that was installed in Multiple Database mode can contain more than one tenant database.

You use this option if a particular schema in a tenant database of the SAP HANA system is to be used as the source system. You can then access tables, views or native DataStore objects of the schema of the tenant database defined in the source system configuration.

You specify the schema as a connection parameter.

> → Tip
>
> You can also create a source system of this connection type directly when creating an Open ODS view.

More information

Related Information

SAP HANA Smart Data Access and SAP HANA Smart Data Integration:

- SAP HANA Smart Data Access in the *SAP HANA Administration Guide for SAP HANA Platform*
- SAP HANA Smart Data Integration
- Administration Guide for SAP HANA Smart Data Integration and SAP HANA Smart Data Quality
- SAP HANA Smart Data Integration and all its patches (SAP HANA SDI 2.0) Product Availability Matrix (PAM)

SAP HANA Tenant Database Schema

For more information about prerequisites and supported database objects, see SAP Note 2312583. - FAQ. BW and SAP HANA Tenant Database: Cross-Database Access in SAP HANA Tenant Databases.

3.12.1.2 Path Prefix and Suffix

For connection type *SAP HANA Smart Data Access*, there are remote sources that provide a hierarchical folder structure instead of a valid database/schema combination in their metadata.

The path is the link for the folder names from the root node to the source object in the structure. The path excludes the remote source (root node) and the source object. It can be divided into a path prefix and a path suffix. The path prefix is the part of the path that is defined in the source system. The path suffix is the remaining part of the path, excluding the source object, which is defined in the DataSource. Both the path prefix and the path suffix can be empty.

The path prefix is a local system setting. It is therefore possible to define a different path prefix in the development system to the one in the quality system or production system. The overview of the objects that can be selected is also improved by restricting it to a part of the folder structure.

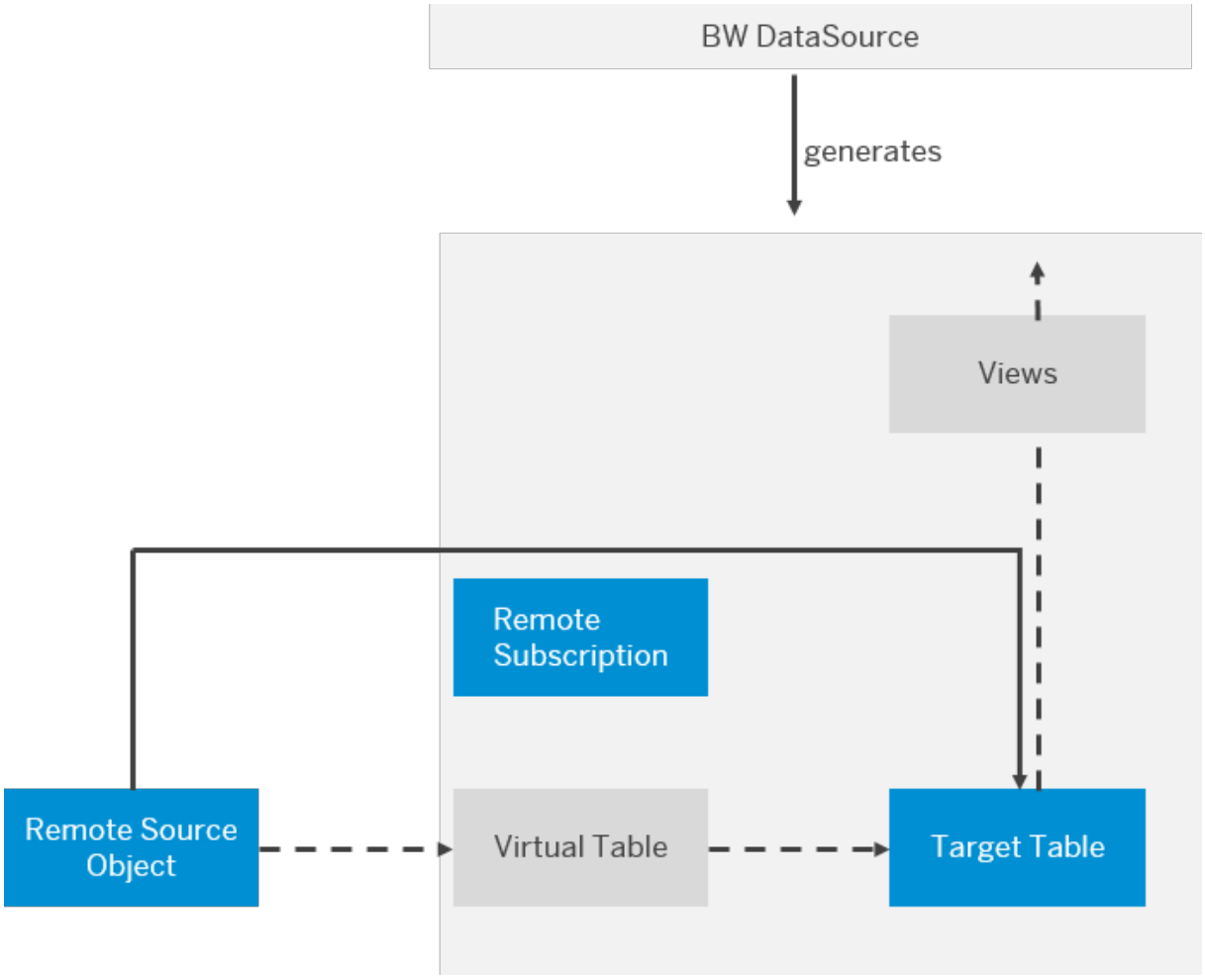
3.12.1.3 Real-Time Replication

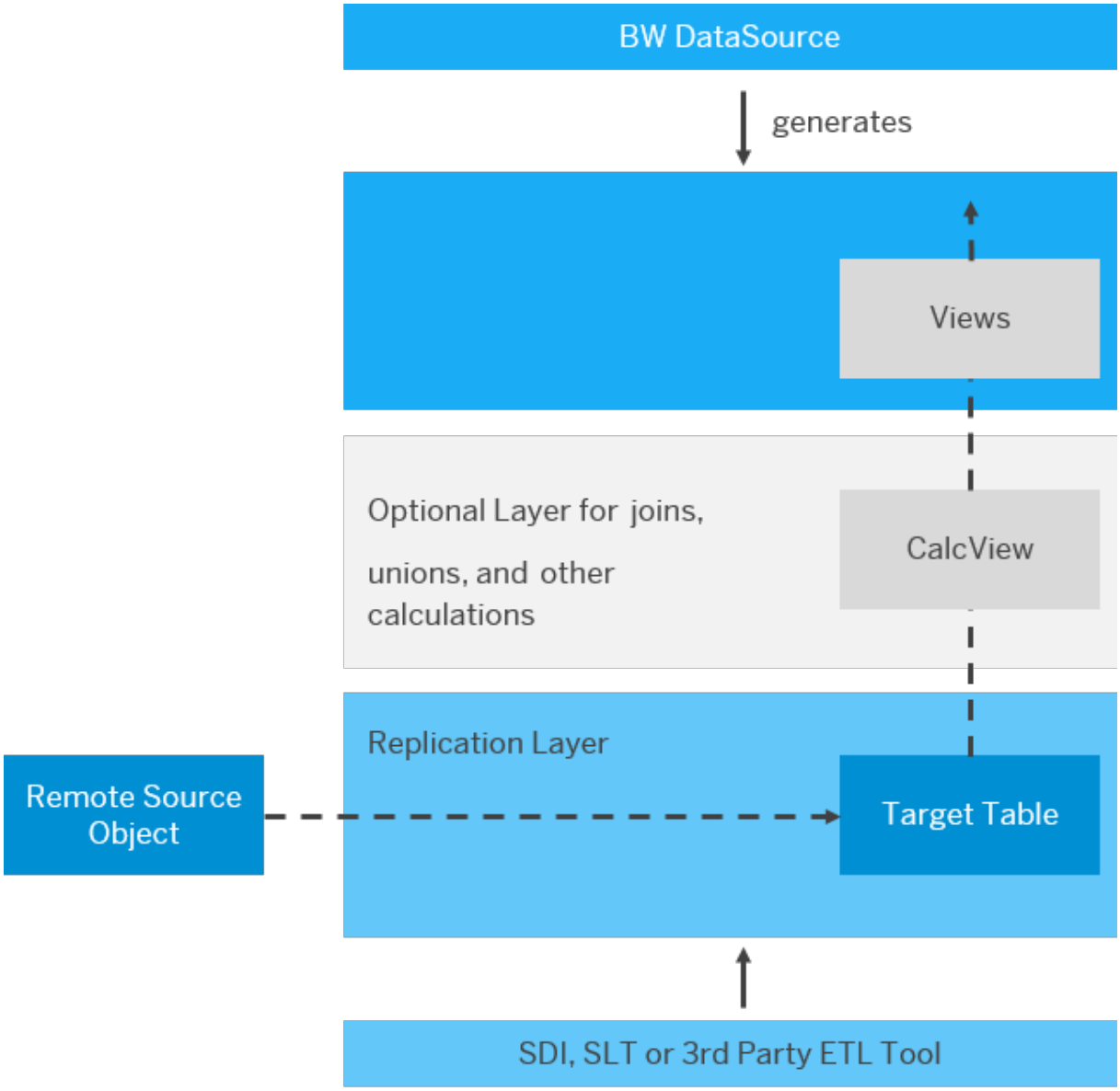
Real-time replication can be implemented with the SAP HANA source system with two approaches or connection types: SAP BW/4HANA-controlled with SAP HANA Smart Data Integration or controlled by an external application replicated to a local SAP HANA database schema.

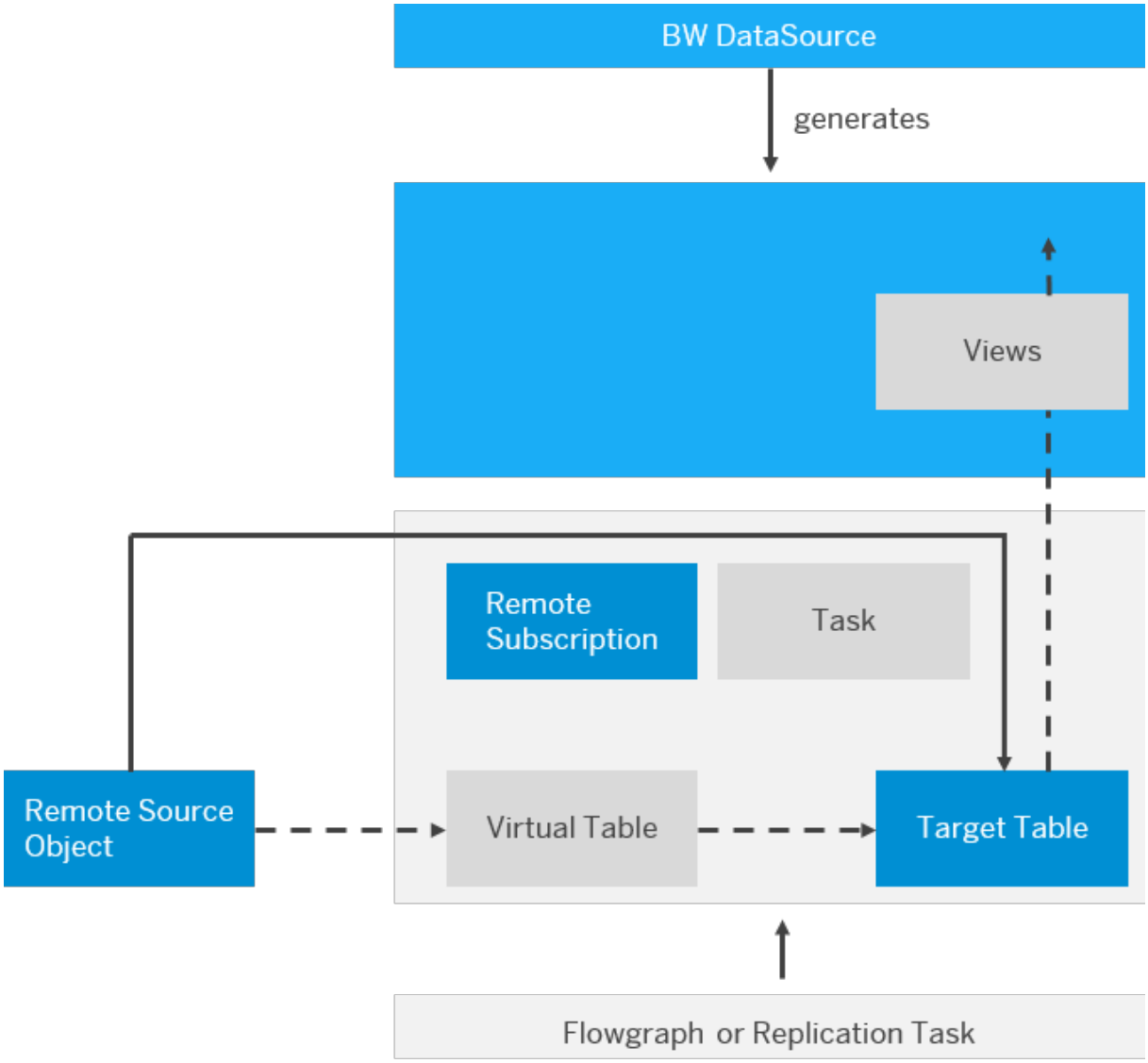
Real-time replication controlled by SAP BW/4HANA is possible using connection type SAP HANA Smart Data Accesss or, to be more precise, SAP HANA Smart Data Integration. SAP HANA Smart Data Integration provides a large number of Smart Data Integration Data Provisioning adapters that support real-time replication. With this approach, SAP BW/4HANA controls the replication process by generating SAP HANA remote subcriptions.

Alternatively, the (real-time) replication process can also be controlled by an external application. This type of application can either be a SAP HANA replication task or a flowgraph in the same SAP HANA, or an SLT replication server or another ETL process. The target table or target view of the replication process is in a local SAP HANA schema connected as a source system to the SAP BW/4HANA system.

The following graphics illustrate the approaches:




BW DataSource
generates
Views
Optional Layer for joins,
unions, and other
calculations
CalcView
Replication Layer
Remote Source
Object
Target Table
SDI, SLT or 3rd Party ETL Tool



3.12.1.3.1 Real-Time Replication (controlled by SAP BW/4HANA)

Using connection type SAP HANA Smart Data Access or, to be precise, SAP HANA Smart Data Integration, you can control real-time replication by SAP BW/4HANA. To do this, you can choose from a large number of Smart Data Integration Data Provisioning adapters that support real-time replication.

Using SAP HANA remote subscriptions generated by the SAP BW/4HANA system, you can replicate data from the SAP HANA source system to the SAP BW/4HANA system in real-time. The data is available for updating to a DataStore object (advanced) or for direct query access via an Open ODS view. There are two types of remote subscription:

- Remote subscriptions of type UPSERT return a view of the actual status of the data (current data) in the target table of the replication in SAP BW/4HANA.

- Remote subscriptions of type INSERT return the complete delta history in the target table of the replication in SAP BW/4HANA.
  With remote subscriptions of type INSERT, delta data (with complete before and after images) can be updated to all types of DataStore objects (advanced).

If the DataSource uses a data provisioning adapter that supports real-time replication, you can define which type or types of remot subscription should be used for real-time replication. You do this in DataSource maintenance on the *Extraction* tab. The selection depends on the required scenario.

Related Information

https://help.sap.com/viewer/p/HANA_SMART_DATA_INTEGRATION
Administration Guide for SAP HANA Smart Data Integration and SAP HANA Smart Data Quality

3.12.1.3.1.1 Remote Subscription of Type UPSERT

Remote subscriptions of type UPSERT provide a view of the current data in the target table of the replication.

A remote subscription of type UPSERT replicates data in real-time from a remote source object to a table on the local SAP HANA database. Type UPSERT means that all types of data changes (inserts, updates, and deletions) are written to the target table with an UPSERT SQL command. For keys that have been deleted from the remote source object, the statement writes the value `D` to column `SDI_CHANGE_TYPE` in the target table. In other cases, the statement writes the value `A` to column `SDI_CHANGE_TYPE`. The commit time stamp of the transaction is written to column `SDI_CHANGE_TIME`.

Use

This type of real-time replication can be used in order to accelerate reporting on an Open ODS view that has a SAP HANA DataSource as its source. It also allows delta extraction and data streaming from a SAP HANA DataSource based on the time stamps in column `SDI_CHANGE_TIME`.

Dependencies

If you select both remote subscription types UPSERT and INSERT at the same time for a DataSource, the delta procedure is determined by remote subscription type INSERT .

If you only select remote subscription type UPSERT for a DataSource, the delta procedure changes automatically to *AIMD After images with deletion indicator*. Three further fields are then added to the DataSource:

- `SDI_CHANGE_TIME`: Returns the commit time stamp of the last data change

- `SDI_CHANGE_TYPE`: Displays the type of the last data change
- `SDI_RECORD_MODE`: Translates the change type into corresponding values of the technical characteristic `0RECORDMODE`

When the DataSource is activated for remote subscription type UPSERT , four additional SAP HANA- objects are created:

- The remote subscription
- A target table for the real-time replication
  The primary key of the target table is the same key as the primary key of the remote source object and the virtual table. The additional columns for the delta extraction are not part of the primary key.
- Two SQL views on the target table – one view for the query access and the full extraction, and another view for delta and delta initialization requests
  The view for the query access and full extraction automatically filters out the deletions (rows with value `D` in column `SDI_CHANGE_TYPE`).

In the target table, the additional columns have the following properties:

- `SDI_CHANGE_TIME` has data type `TIMESTAMP`.
- `SDI_CHANGE_TYPE` has data type `NVARCHAR(1)` and value `D` for deleted record, and `A` (as in Autocorrect) for inserted or updated records.
- `SDI_RECORD_MODE` is not part of the target table. It is calculated in the generated SQL view.

In the SQL view, the additional columns have the following properties:

- `SDI_CHANGE_TIME` is converted into a `NUMC(21)` value by removal of the decimal point from the `DECIMAL(14,7)` display. The time stamp 2015-11-11 11:11:11, for example, which is displayed in SAP HANA as decimal value 20151111111111.0000000, is converted into the numerical character value '20151111111110000000' so that it can be consumed more easily in ABAP.
- `SDI_CHANGE_TYPE` is passed on as `CHAR(1)`. As already mentioned however, value `D` only occurs in the view for delta extraction, as deleted records are not relevant for query access and full extraction.
- `SDI_RECORD_MODE` is calculated for the `SDI_CHANGE_TYPE` value `D` as `0RECORDMODE` value `D` (delete image) and as `' '` (blank, after-image therefore) for all other `SDI_CHANGE_TYPE` values.

3.12.1.3.1.2 Remote Subscription of Type INSERT

Remote subscriptions of type INSERT return the complete delta history in the target table of the replication.

A remote subscription of type INSERT replicates data in real time from a remote source object to a table on the local SAP HANA database. Type INSERT means that all types of data changes (inserts, updates, and deletions) are written to the target table with an INSERT SQL command. The statement writes the original change type to the `SDI_CHANGE_TYPE` column. For each data change, a commit time stamp is written to the `SDI_CHANGE_TIME` column and a unique sequence number is written to the `SDI_CHANGE_SEQUENCE` column.

Use

This type of real-time replication can be used to track all data changes that a remote source object has experienced. In this way, you can, for example, process the complete data change history. Another advantage

of the INSERT remote subscription type compared to the UPSERT type is that before and after images are recorded for each data change. This makes it possible to transfer the data changes to an InfoCube-like INSERT-only target or to aggregate the key figures during the transformation with the "Summation" aggregation behavior. The data target does not, therefore, have to have the same key as the remote source object. Neither are possible with remote subscription type UPSERT, which only delivers after images with deletion indicator.

Dependencies

If you select the INSERT remote subscription type for a DataSource, the delta process of the DataSource changes automatically to *ABR Complete delta update with deletion indicator*. Five further fields are then added to the DataSource:

- `SDI_CHANGE_TIME`: Returns the commit time stamp of the last data change
- `SDI_CHANGE_SEQUENCE`: Returns the sequence number for each data change. The sequence number, together with the commit time stamp, identifies each record uniquely.
- `SDI_CHANGE_TYPE`: Displays the type of the last data change. For updates, two records are added: one with the values prior to the change and one with the values after the change.
- `SDI_RECORD_MODE`: Translates the change type into corresponding values of the technical characteristic `0RECORDMODE`. For updates: a before image and an after image
- `SDI_ENTITY_COUNTER`: Allows a distinction to be made between rows with zero values in the remote source object and rows that have been deleted from the remote source object

When the DataSource is activated for remote subscription type INSERT, three additional SAP HANA objects are generated:

- The remote subscription
- A target table for the real-time replication
  The primary key for the target table is `SDI_CHANGE_TIME` and `SDI_CHANGE_SEQUENCE`, since the primary key values of a record that has been changed in the remote source object are contained multiple times in the target table.
- An SQL view on the target table

In the target table, the additional columns have the following properties:

- `SDI_CHANGE_TIME` has data type `TIMESTAMP`.
- `SDI_CHANGE_SEQUENCE` has data type `INTEGER`.
- `SDI_CHANGE_TYPE` has data type `NVARCHAR(1)` and value `I` for inserted records, value`D` for deleted records, value `B` for values prior to an update, and value `U` for values after an update.
- `SDI_RECORD_MODE` is not part of the target table. It is calculated in the generated SQL view.
- `SDI_ENTITY_COUNTER` is not part of the target table. It is calculated in the generated SQL view.

In the SQL view, the additional columns have the following properties:

- `SDI_CHANGE_TIME` is converted into a `NUMC(21)` value by removal of the decimal point from the `DECIMAL(14,7)` display. The time stamp 2015-11-11 11:11:11, for example, which is displayed in SAP HANA as decimal value 20151111111111.0000000, is converted into the numerical character value '20151111111110000000' so that it can be consumed more easily in ABAP.
- `SDI_CHANGE_SEQUENCE` is passed as `INT4`.
- `SDI_CHANGE_TYPE` is passed as `CHAR(1)`. However, for values `B` and `D`, the signs for key figures are reversed with aggregation behavior "Summation".

- `SDI_RECORD_MODE` is calculated for the `SDI_CHANGE_TYPE` value `B` as `0RECORDMODE` value `X` (before image), as `R` (reverse image) for the `SDI_CHANGE_TYPE` value `D`, and as `' '` (blank, that is, after image) for all other `SDI_CHANGE_TYPE` values.
- `SDI_ENTITY_COUNTER` is calculated for the change type values `B` and `D` as `-1`, and for all other change types as `1`.

3.12.1.3.1.3 Key Figure Summation for Before Images Without After Images

If you use the *Complete delta update with deletion flag* (ABR) delta process to transfer data changes (delta) from an SAP HANA source system to a DataStore object with change log, note the following for the transformation. If you select the "Overwrite" aggregation type for all key figures and do not select the "Summation" aggregation type for any key figure, the before images are filtered out in the transformation and only deletions and after images are taken into account. This applies even if a before image is not followed by an after image. For extractors in SAP Business Suite, this case does not occur in the "Complete delta update with deletion flag" (ABR) delta process. This can occur however occur if database tables are used as a source, as illustrated in the following example.

A table has 3 fields (A, B, and C), whereby A is the only key field, B is a selection field, and C is a key figure:

Column A	Column B	Column C
X	X	10
Y	Y	10

The following SQL command changes columns A and C of the first row to 'Z' and 17:

Sample Code

```
update example set a = 'Z', c = 17 where a = 'X';
```

The corresponding before image and after image have different keys:

Column A	Column B	Column C	Change Type
X	X	-10	Before image
Z	X	17	After image

The following SQL command changes columns B and C of the second row to 'Z' and 18:

Sample Code

```
update example set b = 'Z', c = 18 where a = 'Y';
```

If a filter that selects values from 'X' to 'Y' only is defined on column B in the DTP, the after image is filtered out and only the before image is taken into account for activation.

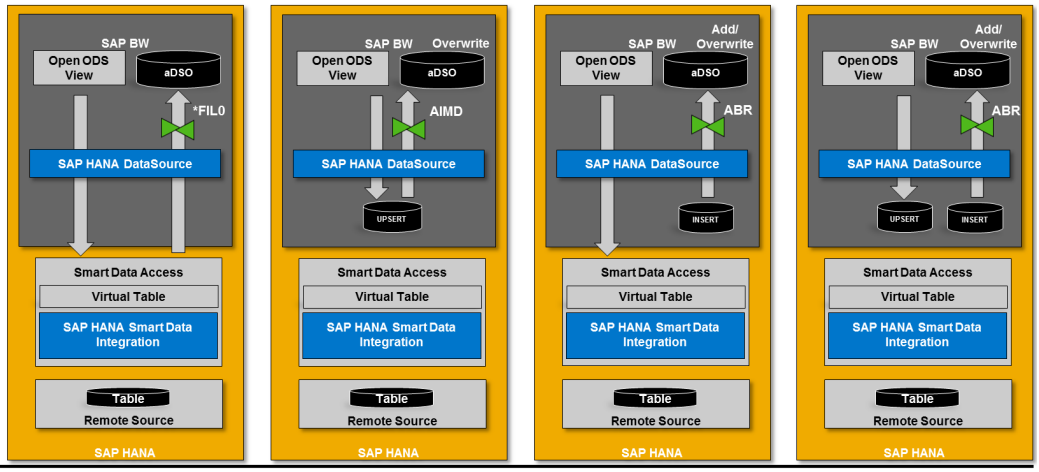
Column A	Column B	Column C	Change Type
Y	X	-10	Before image
Y	Z	18	After image

In each case, the activation result does not correspond to the expected result or to the result that would be obtained using the "Summation" aggregation type.

For this reason, choose "Summation" for all key figures if you are not sure whether before images without after images could arise as in the example shown. Note that this recommendation applies only if you are transferring data from an SAP HANA source system to a DataStore object with change log using the *Complete delta update with deletion indicator* (ABR) delta process.

3.12.1.3.1.4 Scenarios for Data Access Using Smart Data Access

The following figure shows the possible scenarios for direct access and data replication if you have set up the source system connection in the SAP HANA source system using SAP HANA smart data access. Depending on the data provisioning adapter used and the use of SAP HANA smart data integration, various options are available for direct access or extraction to data targets.



Extraction	Virtual Table	UPSERT Table	INSERT Table	INSERT Table
Direct Access	Virtual Table	UPSERT Table	Virtual Table	UPSERT Table

3.12.1.3.1.5 Starting an SAP HANA Remote Subscription

For a DataSource with real-time replication, you can perform the delta initialization, that is, start the remote subscription, the initial data transfer, and the data replication.

1. You are in the BW modeling tools in the DataSource on the *Extraction* tab page.
2. For each remote subscription selected under *Real-Time Replication*, choose *Manage*. The administration screen of the remote subscription for the DataSource appears.
3. To start the delta initialization, choose *Execute*.
4. Specify whether you want the delta initialization to be started in dialog mode or in the background. If it is executed in the background, you can call the job log by choosing the *Jobs* pushbutton.

Executing the Delta Initialization Using a Process Chain

You can add the delta initialization to a process chain using the *Start SAP HANA Remote Subscription* process type. In this case, you specify in the process variant maintenance how the system is to behave in the event of a failed or repeated delta initialization for a remote subscription. If the remote subscription has a status that describes an intermediate state (for example, MAT_START_BEG_MARKER or AUTO_CORRECT_CHANGE_DATA), the system resets the remote subscription by default and deletes the data from the target table. If the remote subscription has an active status (APPLY_CHANGE_DATA), the system terminates the process with a success message by default.

For information about the remote subscription status, see *Remote Subscription States for Data Provisioning* in the *Administration Guide for SAP HANA Smart Data Integration and SAP HANA Smart Data Quality*.

Related Information

Administration Guide for SAP HANA Smart Data Integration and SAP HANA Smart Data Quality

3.12.1.3.1.6 Administration and Monitoring of Remote Subscriptions

Administration of Remote Subscriptions

By choosing the *Manage* pushbutton for a selected remote subscription of a DataSource with real-time replication, under *Real-Time Replication* on the *Extraction* tab page, you can open the administration screen for the remote subscription.

You can select the following functions here:

- You can perform a delta initialization with or without data transfer for the remote subscription. More information: Starting an SAP HANA Remote Subscription [page 154]
- If you have added the delta initialization to a process chain, you can call process chain maintenance by choosing the *Process Chain* pushbutton in the toolbar.
- You can check the current status of the remote subscription by choosing the *Check* pushbutton. The status is then displayed above the pushbutton.
- You can reset the remote subscription and delete the data in the target table by choosing the *Reset* pushbutton in the toolbar. This is relevant, for example, if you have a view as a source and have added new columns to this view so that you can perform a new delta initialization.
- You can call an overview of all remote subscriptions in SAP BW/4HANA by choosing the *Overview* button in the toolbar.

Overview of the Remote Subscriptions in SAP BW/4HANA

In the *Overview SAP HANA Remote Subscriptions*, information about the remote subscriptions is displayed in a table.

By clicking an entry in the *Exceptions* column, you can display the corresponding error message. By choosing the *Subscription Monitor* pushbutton in the toolbar of the overview, you can open the data provisioning remote subscription monitor in SAP HANA, where you can analyze and monitor the error further.

> **Note**
>
> With SAP HANA Smart Data Integration and SAP HANA Smart Data Quality Version 2.0 SP 02 Patch 2 (2.2.2) for SAP HANA 2.0 SPS 02, the data provisioning remote subscription monitor in the SAP HANA cockpit can display the DataSource name and source system name for a remote subscription in column *Design Time Name* in table *Remote Subscription Monitor*. For DataSources activated in SAP BW/4HANA 1.0 SPS04 or higher, this information is displayed automatically. For DataSources activated in earlier releases, this information can be added by re-activating the DataSource.

You can also see from the *Last Message Applied* column whether there is a problem with replication with the remote subscription. This shows when the last change was transferred. A comparison between the virtual table and the target table of the remote subscription can also be helpful in analyzing errors.

For more information about remote subscription monitoring, see *Administration Guide for SAP HANA Smart Data Integration and SAP HANA Smart Data Quality*.

Related Information

Administration Guide for SAP HANA Smart Data Integration and SAP HANA Smart Data Quality

3.12.1.3.1.7 Reorganizing Target Tables for Real-Time Replication

Target tables for real-time replication (delta tables) grow continuously so long as the remote subscription remains active and has not been reset. By reorganizing, you can set a minimum retention period after which data changes that have been successfully fetched by all data transfer processes extracting from the target tables will be compressed.

Context

No records are deleted from the target tables in real-time replication with remote subscriptions of type INSERT or replication with remote subscriptions of type UPSERT. It is therefore necessary to regularly perform the housekeeping task of compressing data changes that are no longer required (delta history). This is particularly important for remote subscriptions of type INSERT, as all data changes are added to the target table here. Housekeeping is also important for remote subscriptions of type UPSERT however, as the number of logically deleted records can increase sharply over time.

Data changes that have been successfully fetched by all data transfer processes extracting from the target tables are generally no longer needed, apart from in the following cases:

- A new data target is added and has to be filled initially.
- An existing data target has to be reloaded.

Reorganisation handles data changes requiring compression in such a way that these cases apply. In other situations, where the data changes are still required, reorganizations should not be used. To prevent unwanted loss of data, reorganization must be started or planned explicitly by an administrator. There is no automatic scheduling or execution by the system.

Procedure

1. Go to *Administration* in the Data Warehousing Workbench and choose *Monitors* > *SAP HANA Remote Subscriptions*.

   The Remote Subscriptions overview screen appears.
2. Choose *Goto* > *Reorganize Delta Tables* from the menu.
3. Specify a minimum retention period in hours.

   The minimum retention period applies for data changes in a target table for real-time replication after all data transfer processes extracting from the table have successfully propagated the data to the connected targets.
4. To perform the reorganization, choose one of the following options:

Option	Description
Programm > *Execute in Background*	If you choose this option, you can set the start conditions and frequency of the reorganization.
(*Quick Schedule*) **button**	You can use this option if no periodic scheduling is displayed in the status row under the minimum retention period, and you have the required authorizations. If you choose this option, reorganization is scheduled daily at 2:00:22 (system time) with the BW background user as the executing user.
Program > *Execute*	For debugging purposes, reorganization can be executed in the dialog.

In the status row, the system indicates whether reorganization is scheduled as a periodic job.

Results

A periodically scheduled reorganization can compress data changes as soon as the minimum retention period has expired.

For remote subscription type UPSERT, the compression deletes the logically deleted records (all records with a `SDI_CHANGE_TYPE` value `D`) that have a `SDI_CHANGE_TIME` value that is before the expiry date.

For remote subscription type INSERT, the compression works differently depending on whether the record was deleted on the expiry date or logically existed on the expiry date:

- Records that were deleted on the expiry date are identified by the last `SDI_CHANGE_TYPE` value before the expiry date being `B` (before image) or `D` (deletion). These records are deleted, together with all older records having the same semantic key.
- Records that logically existed on the expiry date are identified by the last `SDI_CHANGE_TYPE` value before the expiry date being `I` (insert) or `U` (update). The `SDI_CHANGE_TYPE` of these records is set to I (insert), and all older records with the same semantic key are deleted.

Next Steps

If reorganization is scheduled periodically, you can monitor execution of the job by pressing the (*Job Overview*) button.

In the logs of completed jobs, you can see how many records have been processed per DataSource. You can also delete released jobs in the Job Overview in order to reset the reorganization scheduling.

3.12.1.3.1.8 Changing the Remote Subscription Type

You can change the remote subscription type for real-time replication for an active DataSource. You can switch from one remote subscription type to another, select an additional remote subscription type, or deactivate real-time replication. Note, however, that such changes can affect data extraction and direct access.

You can change the selection for real-time replication for an active DataSource in the DataSource editor if the data provisioning adapter used by the DataSource supports real-time replication. You can use the *Change*

Remote Subscription Type pushbutton on the *Extraction* tab page to do this. When you choose this pushbutton, a wizard opens in which you make the change in the first step. In the next step, the wizard shows the effects of the change on an existing delta extraction (data transfer process) or direct access (Open ODS view) if the DataSource is activated with the changes. The change is not made until you have confirmed this step by choosing *Finish*.

Changing the remote subscription type can result in changes to the delta process of the DataSource. This affects the data transfer process and, therefore, data extraction. Changing the remote subscription type can also result in data access to another table of the DataSource. This means that another table is accessed directly by Open ODS views.

Example: Changing remote subscription type UPSERT to remote subscription types UPSERT and INSERT

Data transfer processes for which the source is a DataSource whose remote subscription type has been changed from UPSERT to UPSERT and INSERT are set to inactive. This is because the delta process of the DataSource is changed from *AIMD After images with deletion indicator* to *ABR Complete delta update with deletion indicator* as a result of the change. To transfer the entire delta history to the data target, you perform a delta initialization. When the data transfer process is reactivated, the system checks whether the changed delta process of the DataSource is compatible with the target of the data transfer process.

Alternative to Changing the Replication Method

As an alternative to changing the remote subscription type, you can create an additional DataSource on the remote source object and connect this DataSource to the same data target. Note that any existing data transfer process in extraction mode Delta on the original DataSource must be terminated to avoid duplicate records in the data target. The switch from the original delta DataSource and original data transfer process to the new delta DataSource and new data transfer process must be made with care to ensure delta consistency in the data target.

3.12.1.3.1.9 Condition for Static Filters for SAP HANA DataSources

If the data provisioning adapter makes this necessary, you can specify a static filter condition with SQL syntax for SAP HANA DataSources that support real-time replication. For most SAP HANA DataSources, you do not have to enter a static filter condition. This condition can no longer be changed once the initial data load to the generated target tables has been completed.

Only use this filter if the remote source makes this necessary, to specify a particular data set or restrict the data volume.

Example

If you create an SAP HANA DataSource to read Tweets using the Twitter adapter, you should enter a filter condition to specify a particular data set. If you do not enter a filter condition for the "Status" table, note that the Twitter adapter will read Tweets from your home timeline only.

To read all Tweets from the 'SAP' user's timeline, use the following filter condition: `"ScreenName" = 'SAP'`

To read all Tweets from the public timeline (search), use the following filter condition: `"Tweet" like '%SAP%'`

3.12.1.3.1.10 Effects of DDL Changes in the Remote Source Object on Real-Time Replication

Various data provisioning adapters in SAP HANA Smart Data Integration support propagation of DDL changes in the remote source to SAP HANA. As of SAP BW/4HANA 1.0 SPS04, remote subscriptions managed by SAP BW/4HANA are created with DDL propagation if certain conditions are met. The target table of the replication and the virtual table are then automatically adapted in accordance with the DDL changes, for example if a column is added to the source. The SAP HANA DataSource can apply the changes and adapt the field list in accordance with the DDL change, without the need to reset real-time replication or reinitialize the delta process. Downstream objects in the data flow, such as transformations, data transfer processes and InfoProviders, must be adapted accordingly.

Background

As of Version 1.3, SAP HANA Smart Data Integration supports propagation of DDL changes in the remote source - for example adding a new column to a remote source - to SAP HANA. Propagation of DDL changes is only supported by certain data provisioning adapters. The DDL changes that an adapter can support depend on the version of the adapter in question. The Oracle Log Reader adapter supports more types of DDL changes in Version 2.0 than in Version 1.3 for example.

You can find information about which data provisioning adapters support which DDL changes during replication in the relevant version of the *Configuration Guide* for *SAP HANA Smart Data Integration and SAP HANA Smart Data Quality*, in the section dealing with Options for Replication Behavior for Objects in Replication Tasks. See the adapter documentation in the Administration Guide for SAP HANA Smart Data Integration and SAP HANA Smart Data Quality for more information.

> **Note**
>
> For some adapters, it is possible to define a DDL scan interval. From the moment the DDL change is made through to the moment when the DDL changes are propagated to the target database, no DML changes are permitted.

Remote subscriptions created by the SAP BW/4HANA system prior to SAP BW/4HANA 1.0 SPS04 cannot propagate DDL changes and terminate if the structure of the remote source object has been modified by a DDL command. To continue real-time replication after this kind of change, delta initialization must be performed

again once the DataSource has been adapted and activated in SAP BW/4HANA. On the other hand, remote subscriptions created by the BW/4HANA system in SAP BW/4HANA 1.0 SPS04 and higher can progagate DDL changes in the remote source object. Once the DataSource has been adapted and activated, it is not necessary to reset the real-time replication or reinitialize the delta.

Prerequisites

If certain conditions are met, SAP BW/4HANA creates remote subscriptions with DDL propagation for the supported adapter types:

- SAP HANA 1.0 SPS12 or higher
- No static filter condition (SQL) is specified in the DataSource.
- All fields in the DataSource are selected for transfer.

System Behaviour with DDL Propagation

For a DataSource with remote subscription with DDL propagation, the following applies: If DDL changes are made in the remote source object, the target tables of the replication and the virtual table are adapted accordingly, subject to the restrictions listed in the *Configuration Guide* for *SAP HANA Smart Data Integration and SAP HANA Smart Data Quality*.

As long as you have not adapted the DataSource in SAP BW/4HANA to the changes, the DDL propagation has no effect on the query-type access and the delta extraction in SAP BW/4HANA. This is because both access types are to database views that are generated when the DataSource is activated with remote subscription(s), and do not change unless the DataSource changes. If you adapt and then activate the DataSource following a DDL change in the remote source object in SAP BW/4HANA, you then have to adapt the objects further on in the data flow accordingly.

Adjusting the DataSource in SAP BW/4HANA

In the development system, you use the *Update DataSource* function in the DataSource Editor to adapt the fields in the DataSource in accordance with the DDL changes propagated by the remote source object to the development system. You use the usual remodeling functions to adapt the relevant transformations, data transfer processes and InfoProviders. Once the DDL changes have been propagated to the quality system or production system, adapt the relevant objects by importing the transport request from the development system.

See also

Configuration Guide for *SAP HANA Smart Data Integration and SAP HANA Smart Data Quality* on the SAP HANA Smart Data Integration and SAP HANA Smart Data Quality page in the SAP Help Portal.

Administration Guide for *SAP HANA Smart Data Integration and SAP HANA Smart Data Quality* on the SAP HANA Smart Data Integration and SAP HANA Smart Data Quality page in the SAP Help Portal.

Metadata Comparison and Updating DataSources [page 497]

3.12.2 Transferring Data with a Big Data Source System

The Big Data source system enables central, unified access to sources, which can be provided via SAP HANA Smart Data Access by using the SAP HANA Spark Controller and the SPARK SQL (DESTINATION) or VORA (ODBC) adapters.

Use

You can access the data of a Big Data source system virtually using Open ODS views and CompositeProviders. Data transfer is supported in full mode for all sources, provided the source supports this (generic delta).

Related Information

Creating Source Systems of Type “Big Data” [page 463]
Creating a Big Data DataSource [page 486]

3.12.3 Transferring Data Using Operational Data Provisioning

You can use the Operational Data Provisioning Framework (ODP) to transfer data from various SAP repositories to SAP BW/4HANA.

Operational Data Provisioning offers the following communication channels for data transfer:

- Remote Function Call (RFC)
  You can access ABAP systems via RFC.
- Web Service (SOAP via HTTP or HTTPS)
  The Web Service implementation can occur outside of ABAP.

The data in the source is available for extraction in the form of operational data providers. An operational data provider defines interfaces for transaction data and master data (attributes, texts, hierarchies). Once implemented for the repository in question, these interfaces make it possible to access data for replication to SAP BW/4HANA. For the SAP repositories, there are corresponding providers for the operational delta queues. The SAP BW/4HANA system that the data is transferred to subscribes to these delta queues. For sources that can deliver data changes, the ODP framework also supports delta procedures. Once an SAP repository has implemented the relevant interfaces of the ODP framework, it is available for data transfer with Operational Data Provisioning.

For more information about operational data provisioning

- Extracting and Replicating Data with Operational Data Provisioning Framework [page 177]
- Operational Data Provisioning (ODP) FAQ on https://www.sap.com/

Process

1. You create an ODP source system by connecting the SAP repository as the source of the data transfer to the SAP BW/4HANA system.
   By specifying a ODP context, you define which SAP repository can be accessed using the source system. You can use the following SAP repositories available:

SAP Repository	ODP Context	Communication via RFC	Communication via HTTP	More Information
SAP systems that provide extractors for the transfer	SAPI (DataSources/Extractors)	x	-	Transferring Data from SAP Source Systems via ODP (Extractors) [page 163]
SAP BW or SAP BW/4HANA systems that provide InfoProviders or export DataSources for the transfer	BW (BW/4HANA or SAP NetWeaver Business Warehouse)	x	-	Transferring Data from SAP BW or SAP BW/4HANA Systems via ODP (InfoProviders, Export DataSources) [page 165]
SAP systems that provide ABAP CDS views for the transfer (for example SAP S/4HANA)	ABAP_CDS (ABAP Core Data Services)	x	x (for SAP S/4HANA Cloud for example)	Transferring Data from SAP Systems via ODP (ABAP CDS Views) [page 168]
SAP systems that use the SAP Landscape Transformation Replication Server to provide tables of supported sources in real-time for the transfer.	SLT~<queue alias| (SAP LT queue alias <queue alias>)	x	-	Transferring Data from SAP Systems with SAP Landscape Transformation Replication Server via ODP (Tables) [page 170]
SAP HANA databases (that an ABAP system runs on) that provide SAP HANA views for the transfer	HANA (HANA information views)	x	-	Notes about transferring data from SAP HANA using ODP [page 173]

SAP Repository	ODP Context	Communication via RFC	Communication via HTTP	More Information
SAP Business ByDesign systems	BYD (ByDesign)	-	x	Notes About Transferring Data from SAP Business ByDesign Systems Using ODP [page 175]

2. You create a DataSource for an operational data provider belonging to the source. ODP DataSources can be created manually or by replication in SAP BW/4HANA.

Result

You can now set a transformation between the DataSource and an InfoProvider (DataStore object) and load the data to the InfoProvider using a data transfer process. Alternatively, if the DataSource supports direct access, you can use it as a source in an Open ODS view.

In the source system, you can monitor the data extraction and data replication in the monitor for operational delta queues.

Note that the implementation of the ODP framework in SAP ByDesign does not use any operational delta queues.

Related Information

Creating Source Systems of Type ODP (RFC) [page 464]
Creating Source Systems of Type ODP (SOAP/HTTP) [page 465]
Creating DataSources for ODP Source Systems [page 487]

3.12.3.1 Transferring Data from SAP Source Systems via ODP (Extractors)

You can use Operational Data Provisioning (ODP) to connect an SAP system, such as an ERP system, as the source system to the SAP BW/4HANA system. Communication is performed using RFC. DataSources are provided for use in SAP BW/4HANA using the ODP context for DataSources (extractors) (*SAP*).

Data provisioning using Operational Data Provisioning offers you the following advantages:

- Communication between source and SAP BW/4HANA system and the configuration of data transfer are simplified. In this way communication using synchronous RFC, ALE/IDocs is skipped and the data trasnfer is not controlled using the qRFC outbound queue.
- The Operational Data Provisioning Framework with the new (operational) delta queues takes over tasks and services that were previously covered by the Persistent Staging Area (PSA). The data transfer process

(DTP) in particular can fetch data directly from the (operational) delta queues. InfoPackages and loading using the PSA are thus obsolete. The monitoring of the PSA is triggered by the delta queue monitor (transaction ODQMON).

- Delta data of the application can be retained for a definable time period in the (operational) delta queue and can then be requested for an according time period again by DTP.
- The Operational Data Provisioning Framework uses a unique technology for different consumers of the source data (for example, SAP BW/4HANA and SAP Data Services).

Prerequisites

- SAP systems that provide DataSources for SAP BW/4HANA using the Operational Data Provisioning Framework must have at least one of the following release levels:

Release Level of Source	Additionally Required SAP Note
SAP NetWeaver 7.0 SPS 24	SAP Note 1521883
SAP NetWeaver 7.0 EHP1 SPS 09	
SAP NetWeaver 7.0 EHP2 SPS 08	
SAP NetWeaver 7.3 SPS 08	SAP Note 1931427
SAP NetWeaver 7.3 EHP1 SPS 05	
SAP NetWeaver 7.4 SPS 02	
SAP NetWeaver 7.5	
SAP BW/4HANA 1.0	
SAP BW/4HANA 2.0	
SAP BW/4HANA 2021	
SAP BW/4HANA 2023	

- The DataSources must be released for data provisioning using Operational Data Provisioning. For information about releasing SAP DataSources and customer-defines DataSources for Operational Data Provisioning, see SAP Note 2232584.

Configuration of Data Transfer

Different settings and functions are required for the data transfer from an SAP source system to SAP BW/4HANA. Customizing for extractors (transaction SBIW) is provided in the source system for this. However, note that the following settings and functions listed here are **not** relevant:

- *Maintain the Control Parameter for Data Transfer* (Table ROIDOCPRMS)
  The following table shows if and how the control parameters of the table ROIDOCPRMS are replaced during the data transfer using the ODP framework.

Data Transfer Process

Control Parameter in Table ROIDOCPRMS	Setting for Transfer Using the ODP Framework
Maximum size of data package	A data package of 50 MB is requested by the extractor. The size of a data package to be transferred can be increased in the definition of the data transfer process on tab page *Extraction*. However, the data package transferred by the extractor is not divided up again for the transfer in order to reach the DTP package size exactly. Instead, the uncompressed size of the data package transferred by the DTP usually corresponds to that of the largest multiplication of 50 MB that fits into the DTP package size.
Maximum number of rows in a data package	Not relevant
Frequency with which the info IDocs are sent	obsolete
Maximum number of parallel processes for data transfer	In the data transfer process, you can define the maximum number of work processes using *GoTo* > *Batch Manager Settings*.
Target system of the batch job	Not available
Maximum number of data packages in a delta request	Not available

- *Check Delta Queue* (transaction RSA7)
  The ODP framework provides (operational) delta queues that you can monitor with transaction ODQMON.

Since the data transfer is not controlled using the qRFC outbound queue when using the ODP framework, you do not need any settings in the QOUT Scheduler (transaction SMQS).

3.12.3.2 Transferring Data from SAP BW or SAP BW/4HANA Systems via ODP (InfoProviders, Export DataSources)

The Operational Data Provisioning framework allows you to make use of the ODP source system to use the same technology for a data mart scenario between SAP BW or SAP BW/4HANA systems and SAP BW/4HANA as you would for providing other SAP source data for the SAP BW/4HANA.

The sending system is referred to as the source BW system or source SAP BW/4HANA system, while the receiving system is referred to as the target system. The availability of InfoProviders or generated export DataSources for InfoProviders as the providers of the operational delta queue and of ODP context *BW* in the source make it possible to make InfoProviders of the source in the target as operational data providers available for the data transfer.

It is important to distinguish between two implementations of ODP context *BW* in the source:

Availabilty of ODP Context *BW* in Source Releases as of SAP NetWeaver 7.4 SPS 05 or SAP BW/4HANA 1.0

Prerequisites

- The source BW system has Release SAP NetWeaver 7.4 SPS 05 or higher. SAP BW/4HANA systems are generally supported as sources.
- The target BW system or SAP BW/4HANA system has one of the following releases or higher:
  - SAP NetWeaver 7.3 SPS 08 and SAP Note1780912
  - SAP NetWeaver 7.3 EhP1 SPS 05 and SAP Note1780912
  - SAP NetWeaver 7.4 SPS 05
  - SAP NetWeaver 7.5
  - SAP BW/4HANA 1.0

Supported InfoProviders

For source BW systems:

- DataStore objects (advanced) and (classic) (with delta extraction)
- CompositeProvider - object type HCPR (full-extraction) Release SAP NetWeaver 7.5 or higher
- InfoCubes (with delta extraction)
- Semantically partitioned objects
- HybridProviders
- MultiProviders
- InfoSets
- InfoObjects
  - Master data
  - Texts
  - Hierarchies
- Query as InfoProvider

Für source SAP BW/4HANA systems:

- DataStore objects (advanced) (with delta extraction)
- CompositeProvider - Object type HCPR (full extraction)
- InfoObjects
  - Master data
  - Texts
  - Hierarchies
- Query as InfoProvider

Availabilty of ODP Context *BW* in Source Releases SAP NetWeaver 7.0 to SAP NetWeaver 7.3 EhP 1

Prerequisites

- The source BW system has one of the following releases or higher:
  - SAP BW 7.0 SPS 36
  - SAP BW 7.0 EhP1 SPS19
  - SAP BW 7.0 EhP2 SPS19
  - SAP BW 7.3 SPS 17
  - SAP BW 7.3 EhP1 SPS 20

  > ⓘ Note
  >
  > See SAP Notes 2403202 - ODP API v1.5 and 2404671 - ODQ Queue Model for BW DataMart.

- The target BW system or target SAP BW/4HANA system has one of the following releases or higher:
  - SAP NetWeaver 7.5 SPS 07
  - SAP BW/4HANA 1.0 SPS 03

  > ⓘ Note
  >
  > See SAP Note 2404673 - ODP API v1.5 in BW DataSource.

Supported InfoProviders

Supports export DataSources (8*) generated for InfoProviders in the supported source BW systems. For export-DataSources generated for DataStore objects or InfoCubes, delta extraction is also supported. Note that hierarchy extraction is only possible as of source Release SAP NetWeaver 7.4 SPS05 or with SAP BW/4HANA (see *Availability of ODP Context BW in Source Releases as of SAP NetWeaver 7.4 SPS 05 or SAP BW/4HANA 1.0*).

Procedure

To set up the data mart scenario, perform the following steps in the target system:

1. Create an ODP source system for the source BW or source SAP BW/4HANAwith ODP context *BW*.
2. In the DataSource tree for the ODP source system, create the DataSource for the InfoProvider of the source BW or source SAP BW/4HANA. You have two options:
   - You can replicate the operational data provider from the source BW or source SAP BW/4HANA to the target system as a DataSource.
   - You can create the DataSource.

Authorizations

During extraction from a SAP BW or SAP BW/4HANA system, the data authorizations for the InfoProviders are checked. For more information, see SAP Note 2425281.

Related Information

Creating DataSources for ODP Source Systems [page 487]

3.12.3.3 Transferring Data from SAP Systems via ODP (ABAP CDS Views)

You can use Operational Data Provisioning (ODP) to connect a SAP system as the source system to SAP BW/4HANA.

The ODP context for ABAP CDS views (*ABAP_CDS*) is used to provide ABAP Core Data Services views (ABAP CDS views) with the corresponding analytics annotations for use in SAP BW/4HANA.

In particular, this allows you to access correspondingly annotated ABAP CDS view from SAP S/4HANA and SAP S/4HANA Cloud. If the ABAP CDS view contains the required annotations, you can use the view for the full extraction and the data access or for the delta extraction.

Prerequisites

SAP systems that provide ABAP CDS views for SAP BW/4HANA using the Operational Data Provisioning Framework must have at least one of the following release levels:

For Full Extraction	For Delta Extraction
SAP NetWeaver 7.4 SPS08	SAP NetWeaver 7.5 SPS05 or higher
SAP NetWeaver 7.5 (We recommend SPS02 or higher)	SAP BW/4HANA 1.0 SPS01
SAP BW/4HANA 1.0	

Analytics Annotations for ABAP CDS Views

The ABAP CDS views allow extraction using the analytics annotations:

Full extraction and query-type access (direct access)

The following annotations make it possible for an ABAP CDS view to be available for full extraction or for direct access:

- @Analytics.dataCategory
- @Analytics.dataExtraction.enabled

Delta Extraction

For delta extraction, the following additional annotations are available:

- Analytics.dataExtraction.delta.byElement.name
- Analytics.dataExtraction.delta.byElement.maxDelayInSeconds
- Analytics.dataExtraction.delta.byElement.detectDeletedRecords

  > ⓘ Note
  >
  > We recommend only using this annotation if there is no other way of determining the deltas from the underlying source tables, and after thorough performance tests have taken place.

- Analytics.dataExtraction.delta.byElement.ignoreDeletionAfterDays

> ⓘ Note
>
> Note that the ODP context *ABAP_CDS* only supports deltas with after-images (delta process AIM and AIMD). These delta procedures do not allow direct loading into a data mart DataStore object. To be able to load into a data mart DataStore object, an intermediary standard DataStore object with change log is required.

For more information about annotations, see Analytics Annotations in the Documentation for SAP NetWeaver 7.5.

Supported CDS Views in SAP S/4HANA and SAP S/4HANA Cloud

For detailed information about ABAP CDS views supported for extraction and replication, see the following documentation:

Source Application	Link to Documentation
SAP S/4HANA Cloud	CDS Views in the documentation for SAP S/4HANA Cloud
SAP S/4HANA	CDS Views in the documentation for SAP S/4HANA

> ⓘ Note
>
> For CDS views that extract data in Full mode, not the following: If the source application supports physical deletion in the source tables of the CDS view (to delete incorrect posting directly in the application for

example), this can result in inconsistencies in SAP BW/4HANA if no corresponding reaction takes place in SAP BW/4HANA modeling. To avoid inconsistencies and to properly take account of the deleted records in the standard DataStore object, select the *Snapshot Support* option in the modeling properties of the DataStore object. For more information, see Modeling Properties [page 71].

Real-Time Replication

Delta extraction can also run as real-time replication in streaming mode if the data transfer process is contained in a process chain in streaming mode. The chain will then start when it is notified by the delta queue (ODQ).

Related Information

SAP How-to Guide - How to use ABAP CDS for Data Provisioning in BW

3.12.3.4 Transferring Data from SAP Systems with SAP Landscape Transformation Replication Server via ODP (Tables)

Using SAP Landscape Transformation Replication Server (SAP LT Replication Server), you can provide real-time data from **tables** of supported sources for SAP BW/4HANA. This is achieved by using the trigger-based replication functionality of the SAP LT Replication Server.

Trigger-based replication (SAP LT Replication Server) allows delta processes to be used for tables that do not have any delta-enabled fields. This allows you for example to use delta processes for DataSources (extractors) without delta logic. If you are working with large master data tables without delta logic that are only changed very infrequently, you can significantly reduce the administration effort for frequent full processes by using trigger-based replication (SAP LT Replication Server) as the transfer method.

> ⓘ Note
>
> Note the licensing conditions when using SAP Landscape Transformation Replication Server.

SAP LT Replication Server offers a useful alternative for data transfer to SAP BW/4HANA in the following cases:

- The tables in the source are simple tables without joins or transformation logic.
- The DataSources (extractors) that you want to replace using the SAP LT Replication Server are DataSources on simple tables/views that do not provide a delta mechanism and only contain minimal extractor logic.

> ⚠ Caution
>
> Note the following points if you want to use this transfer method to replace extractors:
>
> - Many extractors contain extractor logic (associations and content logic) that is very complicated to reconstruct in the SAP BW/4HANA system.
> - Many extractors can guarantee a consistent delta transfer to the SAP BW/4HANA system by using the delta queue.
>
> Therefore we recommend that you only replace complex extractors with SAP LT Replication Server in exceptional cases.

You can use operational data provisioning as an interface for implementing the SAP LT Replication Server for transferring data:

- Data transfer using operational data provisioning is supported for tables from SAP systems.
  SAP LT Replication Server provides the operational data provisioning infrastructure with the source tables as delta queues. The data from the delta queue can be replicated to the SAP BW/4HANA system as a subscriber.
  The data is loaded with a data transfer process directly into the InfoProviders. The ODP infrastructure (with delta queues) takes over important services such as monitoring data requests. In addition, the ODP infrastructure has been prepared to support SAP BW/4HANA and other subscribers with SLT data transfer (for example, SAP Data Services).
  .

Related Information

Transferring Data from SLT Using Operational Data Provisioning [page 171]

3.12.3.4.1 Transferring Data from SLT Using Operational Data Provisioning

Configure the connections for transferring data between the relevant systems. In the SAP BW/4HANA system, create the objects required (DataSource and data transfer process) for data transfer.

Prerequisites

- Note the licensing conditions when using SAP LT Replication Server.
- SAP LT Replication Server (SLT) has the minimum release version:
  - Add-On DMIS 2011 SP05
  - SAP NetWeaver 7.3 SPS10, 7.31 SPS09 or 7.4 SPS04 (ODP infrastructure)
- Add-On DMIS_2011 SP03/SP04 or higher or 2010 SP08/SP09 is installed in the SAP source. You also need SAP Note 1863476. If you are using Add-On DMIS 2011 SP05, you need SAP Note 1817467.

- The data transfer is possible for simple tables.

Context

Using the operational data provisioning infrastructure and trigger-based replication of SAP Landscape Transformation Replication Server (SAP LT Replication Server), you can transfer data from SAP systems to the SAP BW/4HANA system in real-time. SAP LT Replication Server is a provider for the operational data provisioning infrastructure. It can make tables from SAP sources available as delta queues. SAP BW/4HANA can be a subscriber to this delta queue and can access the data replicated using the operational data provisioning infrastructure for further processing.

Procedure

1. In SAP LT Replication Server, use transaction `SM59` to create an RFC destination for the SAP system, which is used as a data source.
2. In SAP LT Replication Server, use transaction `LTR` to create a configuration.

   In the configuration, enter the details for the source and target of the SAP LT Replication Server. The source is the SAP system to be used as a data source. The target for data transfer using ODP is the SAP LT Replication Server itself. The connection to SAP BW/4HANA is established later on by defining an ODP source system in SAP BW/4HANA.

   Specifying the source:

   a. Enter the RFC destination for the SAP system to be used as the source.

   Specifying the target:

   a. Enter the **`NONE`** as the RFC connection.
   b. Choose **`ODQ Replication Scenario`** as the scenario for RFC communication.

      Using this scenario, you can specify that data is transferred using the operational data provisioning infrastructure with operational delta queues.

   a. Assign a six-digit queue alias.

      The queue alias is used in SAP BW/4HANA when the source system is created as an ODP context.
3. Create a source system of type ODP in the SAP BW/4HANA system.

   For more information, see .
4. Create a DataSource in the SAP BW/4HANA system.

   For more information, see Creating DataSources for ODP Source Systems [page 487].
5. To start a data transfer, create an InfoPackage or a data transfer process (DTP) in the SAP BW/4HANA system and schedule the data transfer:

Results

The InfoPackage or data transfer process creates a request, which requests the data and loads it into SAP BW/4HANA.

Next Steps

On SAP LT Replication Server, you can monitor the data request by using the delta queue monitor (transaction ODQMON). Select the queue using the provider **`SLT_REPLICATOR`**. Select the subscriber type using subscription **`SAP BW`**and select the queue name using **`<Queue Alias>~<Table Name>`**. To obtain detailed information on the data request (at data data record level), you can navigate through the views of the delta queue monitor.

In the SAP BW/4HANA system, use the extraction monitor or DTP monitor to check whether the data request was completed.

Related Information

Monitoring Delta Queues [page 178]

Information on installation and data transfer with the SAP LT Replication Server can be found under http://service.sap.com/instguides -> *SAP Components* -> *SAP Landscape Transformation Replication Server* -> *Operational Data Provisioning with SAP LT Replication Server*.

3.12.3.5 Transferring Data from the SAP HANA Database in an ABAP System Using ODP (SAP HANA Views)

You can use Operational Data Provisioning (ODP) to connect the SAP HANA database of an ABAP system as the source system to the SAP BW/4HANA system. Communication is performed using RFC. With the ODP context for SAP HANA (*HANA*), analytic views, calculation views and associated attribute views are provided for replication into SAP BW/4HANA. This allows you to perform calculations in the SAP HANA database on the views, before they are loaded into SAP BW/4HANA. These calculations could be currency translations and restricted key figures or key figures calculated before aggregation for example, or joins and unions.

The data transfer with the SAP HANA source system is based on SAP HANA tables however. For more information, see Transferring Data with an SAP HANA Source System [page 142]

Selecting SAP HANA Views via Operational Data Providers in DataSource Maintenance

In the DataSource maintenance transaction, use an operational data provider to select which SAP HANA view you want to transfer the data to the SAP BW/4HANA system from. You can tell which SAP HANA modelling

view can be selected from the name and description of the operational data provider (see below in *ODP Naming Conventions and Length Restrictions for Technical Names*).

Operational data providers are determined from SAP HANA views as displayed in the table:

SAP HANA-View	Semantics of the operational data provider (describes the type of the data to be transferred)	Restrictions/Notes
Analytic View	Transaction Data/Facts	• Calculated columns in the analytic view are only taken into account during extraction if they calculate before aggregation (setting in SAP HANA Modeler). • Attributes in an attribute view that is linked with analytic view's Data Foundation (navigation attributes of the analytic view) are ignored during extraction.
Calculation View	Transaction Data/Facts	• Calculated key figures in the calculation view are only taken into account during extraction if they calculate before aggregation (setting in SAP HANA Modeler).
Attribute View	Master Data/Attributes	-
The description mapping for an attribute in a calculation or attribute view is used to determine a text ODP.	Texts	-

> **Note**
>
> Transferring data from the SAP HANA database with ODP is not supported for hierarchies in attribute views.

ODP Naming Conventions and Length Restrictions for Technical Names

ODP Names:

ODP Descriptions:

The descriptions of the ODPs are taken over from SAP HANA. If there is no description for a SAP HANA view in SAP HANA, the BW system uses <package name>/<name of the view in SAP HANA> as the description.

Authorizations

In the SAP HANA database, analytic privileges are used to define which users are allowed to view which data records for analysis purposes. Analytic privileges are persisted on the SAP BW/4HANA system and in the SAP HANA database.

The SAP HANA database is accessed using the SAP BW/4HANA database user (SAP<SAPSID>). To access a view in SAP HANA, this user requires the corresponding analytic privileges and the authorizations for the corresponding schema. In the SAP BW/4HANA system (ABAP), the access by the user performing the extraction is controlled using authorization object S_RS_ODP_H. The authorization object can control access at SAP HANA package level and at view level. We recommend using the authorization object. If the user performing the extraction does not have authorization object S_RS_ODP_H in his/her profile however, s/he can still perform the extraction if there is a user with the same name with the required analytic privileges in SAP HANA.

To analyze the data replicated into SAP BW/4HANA, the user executing the query requires the corresponding analysis authorizations. You can find out more about defining and assigning analysis authorizations under .

3.12.3.6 Transferring Data from SAP Business ByDesign Systems Using ODP

You can use Operational Data Provisioning (ODP) to connect SAP Business ByDesign system as the source system to the SAP BW/4HANA system. The communication is via HTTP/SOAP (Web Service). With the ODP context for SAP Business ByDesign (*BYD*), analysis views, SAP Business ByDesign data is provided for replication into SAP BW/4HANA. If you are using SAP Business ByDesign in the cloud for subsidiaries of your company, you can integrate local data from SAP Business ByDesign systems and business data from SAP ERP in SAP BW/4HANA for analysis purposes.

Prerequisites for Creating an ODP Source System

1. To configurate the connection in the SAP BW/4HANA system, you need the following information from SAP Business ByDesign:
   - WSDL (Web Services Description Language) file
     This file is needed in order to set up communication between SAP Business ByDesign and the SAP BW/4HANA system. When creating the ODP source system in SAP BW/4HANA, you specify this WSDL file in the SOA Manager in order to set up the Web service.
   - Certificates for a secure connection between SAP Business ByDesign and SAP BW/4HANA:
     The certificates need to be imported into the SAP BW/4HANA system using the Trust Manager (transaction STRUST) before you start creating the ODP source system.

- Depending on the authentification method that you have agreed with SAP Business ByDesign, you need the following information about the credentials for the user who communication is set up for:

Authentification method	Information
SSL Client Certificate	PKCS#12 key pair file with corresponding password This PKCS#12 key pair file needs to be imported into the BW system using the Trust Manager (transaction STRUST) before you start creating the ODP source system.
User ID and password	User ID and password

2. Make sure that ICF service APPL_SOAP_MANAGEMENT is active.
   For more information, see *Activating/Deactivatng ICF Services* on the SAP Help Portal Page for SAP NetWeaver 7.5.

For detailed information about the prerequisites, see the SAP Business ByDesign documentation Integrating Analytics with an SAP NetWeaver Business Warehouse system.

Web Service Configuration in the SOA Manager

In the Web Service Configuration in the SOA Manager, you create and configure a logical port for the default consumer proxy CO_RSDSPX_ODP_IN.

For SAP Business ByDesign systems, once you have entered a name and description for the new logical port, select *WSDL-Based Configuration* as the configuration type and enter the WSDL access setting and the WSDL location. Enter the user name and password under *Configuration of the Logical Port* on the *Consumer Security* tab.

For more information, see *Configuring the Consumer Proxy* on the SAP Help Portal Page for SAP NetWeaver 7.5.

Comments

Note the following:

- SAP ByDesign does not return deltas. The data can be transfered to the SAP BW/4HANA system in extraction mode Full.
- The implementation of the ODP framework in SAP ByDesign does not use any operational delta queues.

More information

For detailed information about the integration of SAP Business ByDesign and SAP BW/4HANA, see Integrating Analytics with an SAP NetWeaver Business Warehouse System.

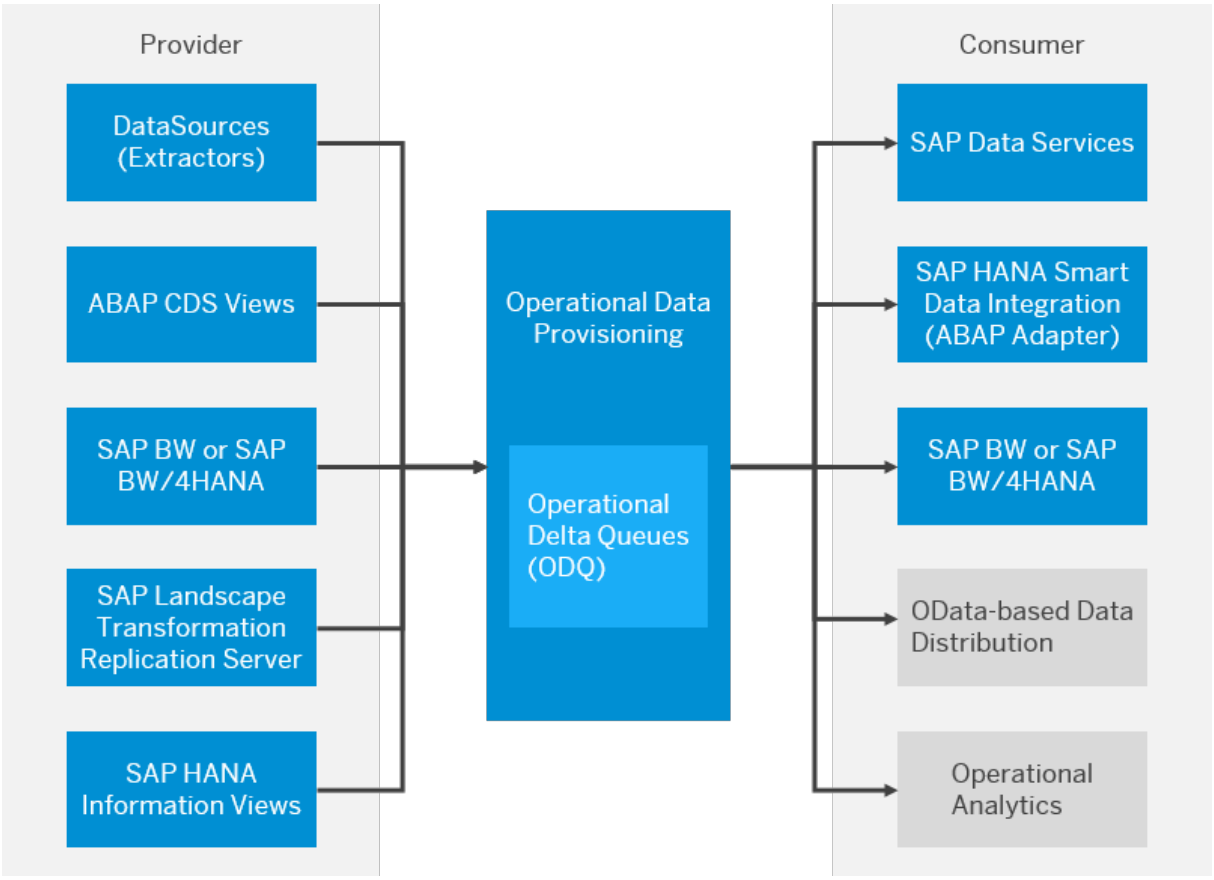
3.12.3.7 Extracting and Replicating Data with Operational Data Provisioning Framework

Operational Data Provisioning (ODP) provides a technical infrastructure that you can use to support two different application scenarios. The first of these is Operational Analytics for decision making in operative business processes. The other is data extraction and replication.

Operational data provisioning supports extraction and replication scenarios for various target applications and supports delta mechanisms in these scenarios. With its service for extraction, transformation and loading (ETL), Operational Data Provisioning supports data transfer to the target applications SAP BW and SAP BW/4HANA, as well as to SAP Data Services or SAP HANA Smart Data Integration (SAP HANA SDI). To support the delta procedure, the data from a source is automatically written using an update process to a delta queue, known as the Operational Delta Queue (ODQ) passed to the delta queue using an extractor interface. Source, also known as (ODP) providers, that can provide the delta queues with data, are at present DataSources (extractors), ABAP Core Data Services Views (ABAP CDS Views), SAP BW or SAP BW/4HANA, SAP Landscape Transformation Replication Server (SLT) and SAP HANA Information Views. The target applications (referred to as subscribers or ODP consumers) retrieve the data from the delta queue and continue processing the data. Monitoring of data extraction and data replication is performed using the delta queues.

In SAP BW/4HANA, Operational Data Provisioning has become the central infrastructure for data extraction and data replication from SAP (ABAP) applications to a SAP BW/4HANA Data Warehouse.

The graphic below provides an overview of the possible and most common ODP integration scenarios:



Related Information

Operational Data Provisioning (ODP) FAQ on https://www.sap.com/

Monitoring Delta Queues [page 178]

3.12.3.7.1 Monitoring Delta Queues

Use

A delta queue is a data store in the source system. Data records are either written automatically to the delta queue using an update process in the source system (for example, 2LIS_11_VAITM) or retrieved using the extractor interface (for example, 0COSTCENTER_ATTR).

The role of a provider is to provide one or more delta queues of a specific type. The "BW DataSource" is an example of a provider. In this case, the delta queue name matches the DataSource name (for example, 2LIS_11_VAITM). The target application of the delta queue is referred to as the "subscriber" (of a specific data services system, for example). A subscription can be defined as follows: a specific subscriber "orders" data changes from one or more queues and continues processing the transferred data. Subscribers are categorized by their subscriber type (for example, SAP BusinessObjects Data Services). A subscription occurs when the subscriber requests data. Every subscription has a unique transaction number, for example {2010-10-22 08:03:52 000001 CET}. A subscriber can have more than one subscription. A queue can also be in multiple subscriptions for the same subscriber.

The data is stored in a compressed state in the delta queue. A *delta* request transfers data records from the queue to the subscriber. The data changes to a queue can also be requested by more than one subscriber. A subscriber can also request data from a queue as a one-off request (Full). In this case, the request is not a subscription.

The data is retained in the delta queue for a specified time period for recovery purposes - in other words, in case the subscriber needs to retrieve the data records again.

Delta Queue Monitor

The Delta Queue Monitor (transaction `ODQMON`) allows you to monitor delta queues in various views:

- Queues
  Here you see all the queues available in the system with the status of each queue, the number of associated subscriptions and the number of associated requests.
- Subscriptions
  The detailed information for the subscription level is displayed here.
- Requests
  The detailed information for the request level is displayed here.
- Units
  The units are displayed here in which the data from a request can be transferred jointly.

You can access the views using pushbuttons or the *Monitor* menu.

Procedure

You are on the Delta Queue Monitor screen (transaction **ODQMON**).

Checking the Data Volume in Queues

The queue overview allows you to identify which delta queues have a high data volume. Queues with a high data volume could mean that requests cannot be retrieved.

1. If you only want to check specific queues, restrict the queue data display by using a preselection in the monitor.
   For more information, see Functions of the Delta Queue Monitor [page 179].
2. To display the data volume of the queues, choose *Calculate Data Volume (Extended View)*.
3. To see which queues contain large volumes of data, you can sort the list of queues by the compressed size in bytes, for example.
4. If you find requests for a queue that have not been retrieved, check these requests. For more information, see Monitoring Requests in the Delta Queue Monitor [page 182].

Checks for When No Data is Delivered to the Subscriber

If no new data from the queue is delivered to the subscriber, check the corresponding requests.

1. Double-click the row with the relevant queue in the overview. The subscription overview for this queue appears.
2. Double-click the row with the relevant subscriber in the overview. The request overview for the selected subscriber in the selected queue appears.
3. Analyze the requests and execute the actions required to continue with the data transfer. For more information, see Monitoring Requests in the Delta Queue Monitor [page 182]

3.12.3.7.1.1 Functions of the Delta Queue Monitor

Use

Restricting the Displayed Information

There are different ways of restricting the information:

Preselection in the Monitor

In the upper area of the monitor screen, you can restrict the data displayed in the queue using various criteria. This improves the performance of the monitor.

- **Provider-Based Restriction**
  If you select a specific provider, for example `BW DataSource`, only the queues belonging to this provider are displayed in the monitor. Subscribers that have not subscribed to any of the queues of this provider are not displayed.
  If you do not select a provider, all the queues from all providers are displayed in the monitor.
- **Queue-Based Restriction**
  If you specify a particular queue, for example, a DataSource, then only this specific queue is displayed in the monitor. When specifying the queue, you can use the wildcard ***** (for example, **0FI***) to restrict the monitor display to several queues.

If you do not specify a queue, the monitor display is not restricted.

- **Subscriber Type Restriction**
  If you select a specific subscriber type, for example `SAP BusinessObjects Data Services`, only the queues that have been subscribed to by a subscriber of this type are displayed in the monitor.
  If you do not select a subscriber type, all the queues of all subscriber types are displayed in the monitor.
- **Subscriber-Based Restriction**
  If you specify a particular subscriber, for example, a data services system, then only this specific subscriber is displayed in the monitor. When specifying the subscriber, you can use the wildcard ***** (for example, **SAP***) to restrict the monitor display to several subscribers.
  If you do not specify a subscriber, the monitor display is not restricted.

These settings are applied when you change between monitor views.

Filter Settings in the List

You can set further restrictions for the displayed data, by using the filter function in the list. Filter settings in the list and changes to the list display only apply to the current monitor view. When you change to a different monitor view, these settings and changes are not applied.

> ### Example
>
> You can restrict the request overview to unconfirmed requests (in other words, requests that have not been transferred by the subscriber).

Calculating and Displaying the Data Volume of Queues

When you *calculate the data volume (extended view)* in the upper screen area of the monitor, the data volume and related key figures are calculated and displayed.

> ### Note
>
> This calculation can take a long time if a lot of units are saved in the delta queues. Therefore the calculation is not performed automatically when the monitor is called. You need to select the relevant checkbox to perform this calculation. When you change to a different monitor view, the calculation setting is applied to the new view.

The following calculations are performed in the monitor view:

- **Queues:**
  For each queue, the system counts the number of units, totals the rows and data volume (before and after compression) and finds the compression rate as well as the smallest and largest sequential transaction number (TSN).
- **Subscriptions:**
  For each subscription, the system counts the number of changed units per queue that are saved (in a retrievable state) in the delta queue. The system totals the changed rows (data changes) and the volume of saved, non-retrieved data (before and after compression), and also finds the compression rate.
- **Requests:**
  For each request, the system counts the number of units per queue, totals the rows and data volume (before and after compression) and finds the compression rate.
- **Units:**
  For a unit, the system totals the rows and the volume of saved, non-retrieved data (before and after compression), and also finds the compression rate. Note that the system always performs the calculation at the units level.

Define Request Selection

In *Request Selection*, in the upper part of the monitor screen, you can set which requests should be taken into account.

As procuring the data for the extended view can be time consuming if you take all requests (with or without subscription), only requests with a subscription (delta requests and delta initializations) are taken into account when the Delta Queue monitor is called. You can also select for example that only real-time requests or only requests without a subscription - requests for one-off transfers of the current data (full) that is - are displayed.

When you change to a different monitor view, the request selection setting is applied to the new view.

Changing Between Monitor Views

You can switch between monitor views using the pushbuttons *queues*, *subscriptions*, *requests* and *units*. You can also double-click on a row to change views:

- **Switch from the queue overview to the subscription overview**
  Double-click on a row in the queue overview. The provider and the queue from the selected row are added to the preselection (in the subscription overview). The subscription overview only displays the subscriptions of the selected queue.
- **Switch from the subscription overview to the request overview**
  Double-click on a row in the subscription overview. The provider, the queue, the subscriber type and the subscriber from the selected row are added to the preselection (in the request overview). The request overview only displays the requests of a specific subscriber in a specific queue.
- **Switch to Unit Overview**
  From all of the other views, you can double-click on the *Units* or *Units (Changed)* column of an entry to switch to the unit overview. Depending on the view you switch from, the provider, the queue, the subscriber type and the subscriber from the selected row in the unit overview are added to the preselection. The unit overview then only displays units for this preselection.

Additional Functions

Additional Monitor Functions

The following additional functions are available in the monitor:

- 
  This function reads the displayed data from the database again.
- 
  This function enables you to choose between the technical names and the queue descriptions in the system language.
- *Goto* > *Reorganize Delta Queues* >
  Use this function to set retention periods for the data in the delta queues and schedule a background job to delete data from the delta queues (after the retention periods have expired).
- *Goto* > *Real-Time Daemon* >
  You can use this function to monitor and schedule the daemon process for real-time processing.
  More information: Monitoring and Scheduling Daemon Processes for Real-Time Processing [page 184]

List Functions

The following list functions are available in the monitor:

-  or 

Use this function (in the queue overview) to find out, for example, which queues contain a lot of data. Once you have calculated the data volume, sort the data records in descending order by the compressed size in bytes.

- 
  Use this function to show or hide columns in a view. You can save these settings, including the sort sequence and filters, as a user-specific layout for a view.

Application Log

In the application log (transaction `SLG1`), you can find logs for extraction requests and aministration tasks, such as terminating subscriptions or the reorganization run. To do this, enter **`ODQ`** and the *Object* and **`Extraction`** or **`Administration`** as the *Subobject*.

3.12.3.7.1.2 Terminating Subscriptions for Inactive Subscribers

Use

If the delta queue records changes for a subscriber who no longer retrieves data, you can terminate this subscription.

Procedure

In the *Subscriptions* view, selecte the entries for the subscription and press .

Result

Once the subscriptions have been terminated, the system displays the requests for the terminated subscriptions in the Request view until such time that the requests are removed from the delta queue during a reorganization run.

3.12.3.7.1.3 Monitoring Requests in the Delta Queue Monitor

Use

A request can be defined as a data request from a subscriber. There are two different types of requests:

- A composite request transfers data from one or more queues that have been grouped together into a subscription.

- An extraction request transfers queue data from the provider to the queue storage.

A composite request can contain several extraction requests. If only data changes are requested that were written to the queue by the source application, the composite request does not contain any extraction requests.

You monitor the requests in the *Requests* overview in the delta queue monitor view. You can perform various actions for the unconfirmed requests (in other words, requests that have not been transferred by the subscriber).

Procedure

Checking the Status of Current Requests

Use the function to check all the displayed requests to see if any have the status or . If the extraction process was canceled or terminated, the status in the database and in the display is updated (corrected).

Investigating Failed Extraction Requests

1. Analyze the cause of the error:
   - Use the function to open the job overview. Here you can display the log of the relevant background job (if it has not been deleted).
   - In the application log (transaction `SLG1`), you can analyze the data extraction logs, using object **`ODQ`** and subobject **`Extraction`**.
   - Use the function to reschedule failed extraction requests. The extraction process is delayed by 60 seconds. This gives you the chance to activate debugging for the background job - in the process overview (function in transaction SM50; can also be called via transaction SM66) or the global work process overview (transaction SM66) - and find the cause of the error.
2. Correct the error.
3. When you transfer data changes (delta), you have the following options:
   - Use the function to repeat the extraction process.
   - Use the function to conclude the corresonding request and set the status of the extraction request to . This status means that the data is transferred again when the next request is made. For more information, see the next section.
4. When you transfer a data snapshot (full), a data request is made again.

Closing Unconfirmed Requests

Use the function to conclude requests that have not been confirmed or canceled by the subscriber. It may be necessary to close a request in the monitor for various reasons:

- The request without subscription has not been retrieved (for example, the connection to target system was deleted).
  In this case, you should not assume that the subscriber will close the request by itself. However, if the request status is not set to or , the delta queue stores the data until the retention period has expired. Completing these types of requests saves memory space and large data snapshots can be removed from the delta queue. The status of the request is set as follows: If extraction fails, the status is set to . If extraction is successful, the status is set to .

- The system could not extract the data changes (delta). (Status:).
  In this case, none of the subscribers to this queue can continue transferring the new data, because data changes must be transferred seamlessly in the correct sequence. Usually, closing the failed request is not enough to continue transferring the data changes for this queue. You need to solve the cause of the problem, otherwise the next extraction attempt will also fail.

  When the request is closed, the extraction request is given status . The request cannot be repeated with the function. The delta process can be continued subsequently by any subscriber. This creates a new extraction request that fits in seamlessly with the most recent successful extraction.

Displaying Units and Data for Specific Queues or Time Periods

You can display units in the *units* overview. The preselection made in the monitor or the navigation route to this view (and the associated time stamps and queues) defines which units are displayed here. Double-clicking a row displays the extracted data for a unit in the bottom of the Monitor screen.

3.12.3.7.1.4 Monitoring Daemon Processes for Real-Time Processing

Use

With the first subscription in real-time mode, with the first connector that you schedule real-time indexing for that is, a daemon process is automatically scheduled in the delta queue (known as the ODQ daemon). If the daemon process is already schedules, it adds new subscriptions to the process in real-time mode. The daemon is automatically removed from scheduling if the last subscription is cancelled, if the associated connector is not scheduled any more for real-time indexing.

Procedure

You monitor the daemon process in the delta queue for real-time processing (ODQ daemon) in the Delta Queue Monitor by choosing *Goto* > *Real-Time Daemon*. You can schedule the daemon manually here if required.

Monitor Daemon Process

A status indicates whether or not a daemon process is scheduled. By pressing , you can call the Job Selection scrren, where all daemon jobs from the last 24 hours are listed. By pressing , you can call the job overview, where you can call each job's job log.

Manually Schedule Daemon Process

If the daemon process had to be stopped, you can schedule it again by pressing with the default settings for Period (15 minutes) and Takt Cycle (15 seconds).

If you want to schedule the daemon process with settings other than the default settings, you can enter the *Period in Minutes* and the *Takt Time in Seconds* and schedule the process by pressing .

3.12.4 Transferring Data from Flat Files

SAP BW/4HANA supports the transfer of data from flat files, files in ASCII format (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange) or CSV format (**C**omma **S**eparated **V**alue).

For example, if budget planning for a company's branch offices is done in Microsoft Excel, this planning data can be loaded into SAP BW/4HANA so that a plan-actual comparison can be performed. The data for the flat file can be transferred to SAP BW/4HANA from a workstation or from an application server.

Process

1. You define a file source system.
2. You create a DataSource in SAP BW/4HANA, defining the metadata for your file in SAP BW/4HANA.
3. You create a data transfer process that includes the parameters for data transfer to the SAP BW/4HANA system.

The metadata update takes place in DataSource maintenance in the SAP BW/4HANA system.

Related Information

Creating Source Systems of Type "File" [page 466]
Creating DataSources for File Source Systems [page 488]

3.12.5 Application Component Hierarchy

The application component hierarchy is used to structure the DataSources in the *Data Sources* tree.

Source systems are displayed in the *Data Sources* tree in the folder for the source system or connection type with an application component hierarchy.

The appearance of the application component hierarchy depends on the type of the source and on the setting made in the source system editor: Certain ODP sources can provide a hierarchical grouping of source objects (tree). For these sources, you can use the *Remote Tree* function in the source system editor to specify whether the hierarchical grouping as an application component hierarchy should be applied to the DataSource tree

for the source system. The following table shows which ODP contexts can provide a hierarchical grouping of source objects and can thus allow a remote tree, as well as from where the remote tree originates:

ODP context	Remote tree possible?	From where does the remote tree originate?
SAPI (DataSources/Extractors)	Yes	The remote tree is taken from the application component hierarchy of the SAP source system (RSA5).
BW (BW/4HANA or SAP NetWeaver Business Warehouse)	Yes	The remote tree is taken from the InfoArea hierarchy of the source BW system.
ABAP_CDS (ABAP Core Data Services)	Yes	The remote tree is taken from the ABAP application component of the package that the CDS view belongs to. Example: CDS view RVPIOBJT > package RSVP > application component BW-WHM-DBA
SLT~<queue alias| (SAP LT queue alias <queue alias>)	No	n/a
HANA (HANA information views)	Yes	The remote tree is taken from the SAP HANA package structure.
BYD (ByDesign)	No	n/a

If you do not want to take a tree from the source, or the source is not able to provide hierarchical grouping (for example the SAP HANA source system or the file source system), the generic application component hierarchy from the SAP BW/4HANA system that is defined for these non-replicating source systems is used in the DataSource tree for the source system. In this generic application component hierarchy, you can create application components and change the hierarchy manually.

In the default setting, empty application components are also displayed. You also have the option of hiding these.

Related Information

Hiding Empty Application Components [page 466]
Creating and Moving Application Components in a Generic Hierarchy [page 467]

3.13 DataSource

A DataSource is a set of fields that provide the data for a business unit for data access and extraction and data transfer into SAP BW/4HANA. From a technical perspective, the DataSource is a set of logically-related fields that are provided for access and transfer data into SAP BW/4HANA in a flat structure (the extraction structure) or in multiple flat structures (for hierarchies).

DataSources supply the metadata description of source data. They are used to extract data from a source system and to transfer the data to the SAP BW/4HANA system. They are also used for direct access to the source data from the BW system.

The following types of DataSource exist:

- DataSource for transaction data
- DataSource for master data
  - DataSource for attributes
  - DataSource for texts
  - DataSources for hierarchies (multiple segments)
- DataSources for segmented data

The data can be loaded into the SAP BW/4HANA system from a source in the DataSource structure using a data transfer process or an InfoPackage. You determine the target to transfer the data from the DataSource to during the transformation. You also assign DataSource fields to InfoObjects or fields of the target object in SAP BW/4HANA here.

> **Note**
>
> For certain types of sources, you can also access the data directly via Open ODS views, without DataSources. The data is also loaded into the SAP BW/4HANA system here using DataSources.

Related Information

Working with DataSources [page 470]

3.14 Transformation

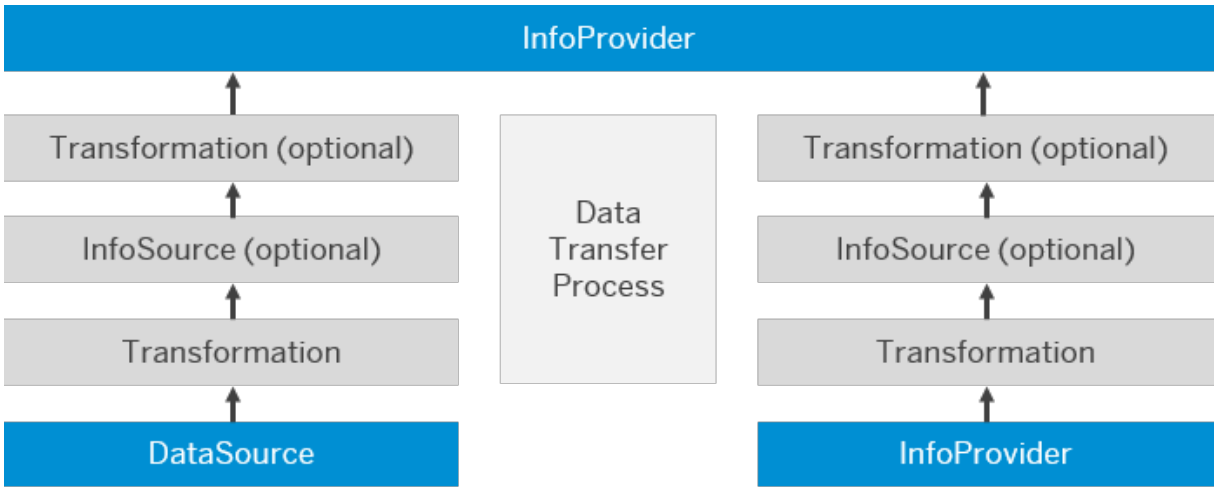
The transformation process allows you to consolidate, cleanse, and integrate data.

You can semantically integrate data from various sources.

You create a transformation between a source and a target. When data is loaded from a BW object or virtual object to a target BW object, the data passes through a transformation. A transformation converts the fields of the source into the format of the target.

A transformation consists of at least one transformation rule. Various rule types, transformation types, and routine types are available. These allow you to create very simple and highly complex transformations.

The following figure illustrates how the transformation is integrated in the data flow:



3.14.1 Rule Group

Use

A rule group is a group of transformation rules. It contains one transformation rule for each key field of the target. A transformation can contain multiple rule groups.

Rule groups allow you to combine various rules. This means that for a characteristic, you can create different rules for different key figures.

Features

Each transformation initially contains a standard group. Besides this standard group, you can create additional rule groups.

If you have defined a new rule in rule details, you can specify whether this rule is to be used as a reference rule for other rule groups. If it is used as a reference rule, then this rule is also used in existing rule groups as a reference rule where no other rule has been defined.

Example

The source contains three date characteristics:

- Order date

- Delivery date
- Invoice date

The target only contains one general date characteristic. Depending on the key figure, this is filled from the different date characteristics in the source.

Create three rule groups which, depending on the key figure, update the order date, delivery date, or invoice date to the target.

3.14.2 Semantic Grouping

If a semantic grouping is used, data records with the same values in the selected fields form a logical key and are processed together in one package.

This setting must be considered in combination with the settings for semantic groups in DTP:

If handling data records with errors is activated, only the sources fields of key fields can be selected in DTP. The actual grouping uses the intersection of these selected key fields and the fields selected from the transformation for semantic grouping.

If handling data records with errors is switched off, all the sources fields that were also selected in the transformation can be selected in DTP. The selection of the semantic group is formed from this intersection.

If handling data records with errors is switched off, and no source fields have been selected in the transformation for the semantic grouping, all the source fields can be selected in DTP. This function has been retained for compatibility reasons, due to it not being possible to display source fields for the semantic grouping in the transformation. We recommend selecting the fields in the transformation however.

Related Information

Conditional Update through Semantic Grouping [page 189]

3.14.2.1 Conditional Update through Semantic Grouping

Use

The purpose of conditional updating is to update data records to a DataStore object only if a certain condition (logical expression) has been fulfilled. The data records should not be updated if this condition has not been fulfilled. To ensure that the most recent data record can always be checked, it must be processed together with all the other data records of the logical key. This can be ensured by using the semantic grouping. Here, all of the data records in a semantic group are processed together in one package.

The conditional update is programmed in the globally valid section of a routine, within the transformation. Two steps are required:

- If the condition has been fulfilled, the source value should be updated.
- If the condition has not been fulfilled, the value in the target is first initialized using the standard setting for the DataStore object. Since this initial value is not to be updated and the original value from the target is to be retained, however, the initial value must be overwritten again with the value from the A table or activation queue.

Further examples of using semantic groupings include the calculation of averages, checksums, and totals across a characteristic or characteristic group.

3.14.3 Error Analysis in Transformations

With the *Simulate and Debug DTP Requests* function, you can simulate a transformation prior to the actual data transfer to check whether it returns the desired results. You can set *Breakpoints* at the following times in the processing: before the transformation, after the transformation, after the start routine, and before the end routine. More information: Simulating and Debugging DTP Requests [page 546].

3.14.4 Authorizations for the Transformation

To work with transformations, you need authorization object S_RS_TR.

3.14.5 Transport the Transformation

The transformation is integrated into BW's TLOGO framework, and can be transported.

Routines are transported together with the transformation.

The transport object is called TRFN.

3.15 Data Transfer Process

A data transfer process (DTP) is an object that determines how data is transferred between two persistent objects (source and target) in the SAP BW/4HANA system.

You use the data transfer process to transfer data in SAP BW/4HANA from one persistent object to another object, in accordance with certain transformations and filters. You can create a transformation between the source and the target of the data transfer process. Alternatively, you can use InfoSources, which do not have persistence, to perform the data transfer process with several consecutive transformations (a transformation path).

The data transfer process makes the transfer processes in the data warehousing layer more transparent. Optimized parallel processing improves the performance of the transfer process (the data transfer process determines the processing mode). You can use the data transfer process to separate delta processes for different targets and you can use filter options between the persistent objects on various levels.

You define data transfer processes in the BW modeling tools. We recommend that you integrate data transfer processes into process chains. In this case, the data transfer process is executed when it is triggered by an event in the predecessor process in the process chain. Alternatively, in process chain maintenance, you can execute a data transfer process in the background. A debug mode is also available.

The request is an instance that is generated at the runtime of the data transfer process. The request is processed in the steps that have been defined for the data transfer process (extraction, transformation, filter and so on). The monitor for the data transfer process request shows the header information, request status, and the status and messages for the individual processing steps.

With a data transfer process, you can transfer data either in full extraction mode or in delta mode. In full mode, the entire dataset of the source is transferred to the target; in delta mode, only the data that was posted to the source since the last data transfer is transferred. The data transfer process controls delta handling and therefore allows you to fill several targets with different deltas from one source. With a data transfer process, you do not need to explicitly initialize the delta method.

3.15.1 Data Transfer Intermediate Storage (DTIS)

The data transfer intermediate storage (DTIS) is a table with technical key. This is generated when a data transfer process (DTP) is activated, or when a SAP HANA transformation is generated for each DTP source object upon generation.

Only one data transfer intermediate storage is generated for an object that is used as a source in multiple DTPs.

The DTIS is used in the following cases:

- For error handling
  If error handling is activated, the DTIS is used as an error stack
- To allow sorting on sources that cannot be read sorted
  For sorting, the key is used that you define in the *Extraction Grouped by* setting (semantic grouping).
- To allow execution of the transformation in SAP HANA if the source does not support this
  SAP HANA DataSources and the DataStore object (advanced) for example support execution of the transformation in SAP HANA. File or ODP DataSources on the other hand do not support SAP HANA execution.
- To allow navigation attributes as keys for the *Extraction Grouped By* setting (SAP Note 3330273)
- To allow the *Extraction Grouped By* setting for both the warm store and the cold store (SAP Note 3137759)
- To improve performance if *Suppress DB Aggregation* is **not** selected (SAP-Hinweis 3316562)
- To process data only once when reading and writing to the same table when using Myself DTPs with extraction mode *Full* (SAP Note 3386485)

The DTIS is based on the data source; it thus stores records from the source.

The system decides whether or not to create a DTIS depending on the DTP source, the DTP target, the transformation, and the settings in the data transfer process. It is created for example for file or ODP source

systems that grouped extraction should be performed for and/or for which the transformations should be executed in SAP HANA.

All records extracted from the source are written to the DITS. They are handled according to the request status, as follows:

- Red: If a DTP request has the status red, all data records in the DTIS are retained.
- Green: If a DTP request has the status green, all data records are normally deleted directly apart from those with errors. If a latency time is set for the Temporary Storage setting for the *Fill Data Transfer Intermediate Storage* process step, you have to schedule program RSBKCHECKBUFFER to delete the records upon expiry of the latency period.
- Deleted When a DTP request is deleted, the associated data records in the data transfer intermediate storage are also deleted.

Data transfer intermediate storage as error stack

The data transfer intermediate storage (DTIS) is used as a persistence layer for error handling in SAP BW/4HANA (error stack). If error handling is activated for the DTP, records with errors are written at runtime to the data transfer intermediate storage. You use the data transfer intermediate storage to update the data to the target destination once the error is resolved.

If error handling is activated, and there are data records with errors, you can call the data transfer intermediate storage and display and edit the data records in question. The authorizations for DTIS maintenance are checked using authorization object S_RS_DTP. You require activity 23 (Maintain DTP Definition). You can call the data transfer intermediate storage in the following ways:

- You can call it in the data transfer process editor by choosing *Open DTIS Maintenance* from the *Update* tab. The data records displayed in the DTIS are then restricted to the current requests for this DTP.
- You can call it from the Request Monitor by choosing *Error Stack*.

With an error DTP, you can update the data records to the target manually or by means of a process chain. Once the data records have been successfully updated, they are deleted from the data transfer temporary store. If there are still any erroneous data records, they are written to the data transfer temporary store again in a new error DTP request.

3.15.2 Handling Data Records with Errors

On the *Update* tab in the data transfer process (DTP), the error handling settings allow you to control how the system responds if errors occur in the data records when data is transferred from a data transfer process (DTP) source to a data transfer process target.

Settings for Error Handling

For a data transfer process, you can specify how you want the system to respond when data records contain errors. If you activate error handling, the data records containing errors are written to a database table, the

data transfer intermediate storage. You can use a special data transfer process, the error DTP, to update the records to the target.

Temporary storages are available after each processing step of the DTP request. This allows you to find out which processing step the error occurred in.

> **Note**
>
> For information about error handling in transformations and data transfer processes during execution in SAP HANA, see SAP Note 2580109.

Checks for Data Records with Errors

The following table provides an overview of where checks for data records with errors can be run:

Where does the check take place?	Examples of Data Records with Errors
In the transformation	Field contains invalid characters or lowercase characters
	Error during conversion
	Error during currency translation
	A routine returns a return code <> 0
	Characteristic value is not found for master data
	Error while reading master data
	Customer-specific formula results in error
When data is updated to the master data table or text table	Invalid characters in key or navigation attribute
	If no SID exists for the value of the navigation attribute
	If no language field is defined for texts
	Implausible "from" and "to" dates
	Duplicate data records (duplicate keys)
	Overlapping and invalid time intervals
When updating data to the DataStore object	If no SID exists for the characteristic value
When checking referential integrity of an InfoObject against master data tables or DataStore objects	If no SID exists for the characteristic value

Data Transfer Intermediate Storage

The data transfer intermediate storage (DTIS) is used as a persistence layer for error handling in SAP BW/4HANA. The data transfer intermediate storage is a table with a technical key, which is generated for each

DTP source object during DTP activation. Therefore only a data transfer intermediate storage is generated for an object that is used as a source in multiple DTPs. If you activate error handling, the data records containing errors are written to the data transfer intermediate storage.

"Repairing" with Error DTP

You create an error DTP for an active data transfer process on the *Update* tab page. You run it directly in the background or include it in a process chain so that you can schedule it regularly in the context of your process chain. The error DTP uses the full update mode to extract data from the data transfer intermediate storage and transfer it to the target, which you specified in the original data transfer process.

3.15.2.1 Temporary Storages

A temporary storage is a table that contains the data records processed with the request in a specific data transfer process processing step. The table contains correct records as well as erroneous records with a status display.

Temporary storages are used at runtime to allow processes to be restarted without a significant impact on performance and consistency when errors occur. They also allow you to find records that have errors. Unlike on the source level, this is only possible if, under Level of Detail, you have set Trace of Erroneous Records or Trace of Transformation by Records.

You make settings for the temporary storage in the data transfer process editor on the *Runtime Properties* tab. For more information, see Specifying How Data Records Are Handled [page 541].

You can display the temporary storage for each of the processing steps using the monitor for the data transfer process request by choosing the pushbutton in the *Data* column. The records with errors can be identified by the symbol.

4 Tasks

The following sections provide a step-by-step explanation of how you can use the BW modeling tools to create and edit objects relevant for Business Warehouse.

4.1 Creating InfoAreas

You create InfoAreas in order to structure objects in the SAP BW/4HANA Metadata Repository.

Create InfoArea on the First Level

1. In the *Project Explorer* view in the context menu of the BW project, choose *New* > *Other*.
2. In the next popup, select the InfoArea Wizard and choose *Next*.
   The wizard for creating InfoAreas appears.
3. To create an InfoArea on the first level of the *BW Repository* tree, leave the *InfoArea* field blank.
4. To ensure that you can find your InfoArea again without difficulty, we recommend setting the *Add to Favorites* flag.
5. Enter a name and a description for your InfoArea and choose *Finish*.

Create InfoArea under an Existing InfoArea

1. In the BW repository tree of the BW project, in the context menu of the InfoArea underneath which you want to create the new InfoArea, choose *New* > *InfoArea*.
   The wizard for creating InfoAreas appears. As an alternative, you can call the InfoArea wizard as described in the previous section.
2. Continue as described above, but leave the target InfoArea as it is in the InfoArea field, or enter a different target InfoArea of your choosing.

Results

The InfoArea is created and activated in the *BW Repository* tree, and displayed if required in *Favorites*. The editor for the InforArea is opened.

4.1.1 Moving InfoAreas and Objects in the InfoArea Hierarchy

In a BW project, InfoAreas and objects can be moved from one InfoArea to another with a native Eclipse editor. Moving to another InfoArea is a transport-related change.

Procedure

1. Select one or more objects and InfoAreas in the *Project Explorer* view in the *BW Repository* tree, or in the *Favorites*, and choose *Move* from the context menu.

   You can also use drag and drop for this. Multiple selection is also supported here.

   The *Move* dialog box appears.
2. Enter a target InfoArea.
3. Specify the position in the hierarchy where you want to insert the InfoArea: as a subnode, above or below the target InfoArea.

   > ⓘ Note
   >
   > The *Above* and *Below* options (above and below the target InfoArea) can only be selected if alphabetical sorting for SAP BW/4HANA objects is deactivated in the BW project. You can find more information in the documentation for the *Display Options for the BW Modeling Tools*.
4. The system displays the list of elements that can be moved. By deselecting elements, you can stipulate that they cannot be moved.
5. Press *OK*.

Related Information

Display Options for the BW Modeling Tools [page 36]
InfoArea [page 40]

4.1.2 Deleting InfoAreas

You can delete empty InfoAreas that have no objects assigned to them and do not contain any further InfoAreas.

Procedure

1. Select an InfoArea in the *Project Explorer* view in the *BW Repository* tree, or in the *Favorites*, and choose *Delete* from the context menu.

2. Confirm the security prompt.

   If the InfoArea still contains elements, the system displays a message informing you that the InfoArea is still in use.

4.2 Working with Data Flows

When working with data flows, you can use various functions.

- **Create data flow objects** in order to depict a scenario with all participating objects and the relationships between them
- **Create documentation for data flow objects and the objects contained in them** in order to document the nodes contained in the data flow object in the context of the scenario
- **Save the entire data flow documentation as an HTML file** (including the metadata descriptions from the metadata repository)
- **Enhanced data flow copy** in order to create multiple target data flows or changes to the copied data flow
- **Create objects and connections of a data flow** using the transient editor (also without a data flow object)
- **Display and analyze the data flow for an object** (using the transient editor)

Related Information

Creating a Data Flow Object [page 197]
Creating Documentation [page 208]
Saving Data Flow Documentation as an HTML File [page 209]
Copying a Data Flow [page 210]
Using a Transient Data Flow for a Persistent Object [page 213]

4.2.1 Creating a Data Flow Object

In a data flow object, you can depict a scenario in a SAP BW/4HANA system, including all participating objects and the relationships between them. When creating a data flow object, you can start from scratch and create non-persistent object together with the connections between them. Alternatively you can create a data flow object in order to depict an existing scenario with persistent and, if relevant, also with existing data flows.

Procedure

1. You are in the BW Modeling tools. Choose *New* > *Data Flow* from the context menu for your BW project on the *BW Repository* node or the required InfoArea.

Alternatively you can create a data flow object in the transient editor. Open the context menu of a SAP BW/4HANA object (in the BW repository, in a data flow object or in the transient editor) and choose *Explore Data Flow*. Next choose *File* > *Save As* in the Eclipse menu.

2. In the dialog for creating the data flow object, make the required entries:
   a. Select an InfoArea.
   b. We recommend selecting the *Add to Favorites* field so that you can call the data flow object quickly next time you need it.
   c. Enter a technical name and a description.
   d. By choosing *Copy from*, you can select an object template. This allows you to copy over the properties from the template to your new data flow object and to edit them as required.
   e. Choose *Finish*.

   The system opens the new data flow object in the native Eclipse Editor. The *General* tab opens.
3. Add persistent objects, non-persistent objects or existing data flows to the data flow object.
4. Define the connections between the objects. Connections can be any of the following:
   - Transformations
   - Data transfer processes
   - Relationships between objects and their constituent parts, for example in an InfoProvider contained in a CompositeProvider
5. [Optional:] Create documentation for the data flow object and its nodes.
6. Save and activate the data flow object.

Related Information

Working with the Data Flow Object Editor [page 198]
Functions in the Data Flow Object [page 201]
Creating Documentation [page 208]
Tips and Tricks for Working with Data Flows [page 214]
Recommendations for Working with Large Data Flows [page 214]

4.2.1.1 Working with the Data Flow Object Editor

The Data Flow Object Editor provides a graphical interface for modeling data flow objects.

Tab: General

On the *General* tab, the system displays teh technical name of the data flow object and the description.

You can change the description here. Under *Add all existing transformations and DTPs automatically*, you can also specify that all transformations and data transfer processes that are not yet contained in the data flow

object but whose source and target objects are part of the data flow object, are now added to the data flow object. Once you have made the setting and have saved and activated the data flow, the transformations and data transfer processes are displayed in the editor.

Tab: Details

On the *Details* tab, the system displays a grid where you can sort and depict the objects and the data flow connections. To add non-persistent objects, you have an object pallet, which you can drag to either the left or the right of the grid.

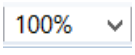
Objects are inserted as nodes in the cells of the grid. Nodes with blue backgrounds represent persistent objects and nodes with white backgrounds represent non-persistent objects. Transformations and data transfer processes are inserted as connections between the nodes. The icons in the target node indicate if the connections are transformations or data transfer processes. The numbers next to the icons indicate the number of connections. To find out if this number includes non-persistent connections, hover with the mouse pointer over the icons. Relationships between objects and their constituent parts, for example InfoProviders contained in a CompositeProvider, are shown by connections (without arrows).

The various functions in the Data Flow Editor can be executed or called using the following UI interactions:

- Drag and drop (from the pallet or from the Project Explorer view onto the grid)
- Context menu (in the grid or on an object)
- Context pad on object

Toolbar

The toolbar in the Data Flow Editor contains the following functions:

Function		Explanation
Functions to display the data flow		You can use the dropdown menu or the slide bar to set the display size.
	 (*Compressed view*)	For a better overview of a larger data flow, you can use the compressed view. The display is reduced in size here, and the nodes show the object type and the technical name of the object only. By choosing  (Decompressed View). you can return to the original view.

Function		Explanation
	 (*Hide connections*)	Data transfer processes and transformations are displayed as symbols in the node of the target object. You can switch off the display of data transfer processes and/or transformations. If both of them are switched off, the connections between the nodes are not displayed any more either.
	 (*Auto Layout*)	For a better overview of a more complex data flow, you can automatically adjust the layout. The system then arranges the nodes and connections in the grid, providing a better overview.
General Data Flow Object Functions	(*Share BWMT Link*)	Using this function, you can either send a link that opens the SAP BW/4HANA object in the native Eclipse editor by e-mail, or you can copy the link to the clipboard and share the link with other people. More information: Sharing Links to SAP BW/4HANA Objects [page 32]
	 (*Show in history view*)	The *History* view displays a list with the various versions of the data flow object, together with the version type and creation date. More information: History View [page 15]
	 (*Delete*)	You can delete the data flow object here.
	 (*Show Documentation*)	You can display the documentation here for an entire data flow in HTML format as a preview, and save the documentation as an HTML file on a computer, or print it. More Information: Saving Data Flow Documentation as an HTML File [page 209]

Important Views for Data Flow Object

View	Meaning for Data Flow Object
Properties	The *Properties* view provides functions that are specific to the Data Flow Object Editor. Here you can: • Display the data flow object metadata • Create and display documentation for the data flow object and its nodes • Display the inbound and outbound connections of a node selected in the editor More Information: Properties View [page 14]

View	Meaning for Data Flow Object
Outline	The *Outline* view lists the nodes contained in the data flow object by object type. It helps you to quickly find a node in the data flow: You can use a filter to quickly find a node in the *Outline* view. If you highlight the node in the Outline view, this node is also highlighted in the editor of the data flow object. The node context menu contains useful functions for opening the documentation, for example, or displaying the connections for the node in the *Properties* view.
Miniature View	The *Miniature View* make it easier to navigate your way though large data flows. This view displays the complete data flow as an overview image and highlights the section, which is currently displayed in the editor of the data flow object. The *Miniature View* is not one of the default views of the BW Modeling Tools. You can show the view by choosing this menu command: *Window* > *Show View* > *Other* > *General* > *Miniature View*.
Problems	The *Problems* view assist you with troubleshooting. More Information: Problems View [page 14]
History	The *History* view displays the available versions of the data flow objects. More Information: History View [page 15]

4.2.1.2 Functions in the Data Flow Object

Nodes

Function	Explanations
Insert non-persistent objects as nodes	You have the following options: • Drag & drop from the pallet • *New Data Flow Node* > *<Object Type>* in the context menu for the grid

Function	Explanations
Replace non-persistent objects with (new or existing) persistent ones	You have the following options: To create a new persistent object: • Double-click on the node for the non-persistent object • *Create Persistent BW Object* in the context menu for the node To replace the non-persistent object with an existing persistent one: • Drag & drop from the Project Explorer view • *Add Object* in the context menu for the node If you insert a CompositeProvider, its components, together with any SAP HANA views, are automatically inserted with it as separate nodes. If the SAP HANA system is linked with the BW project, you can open the SAP HANA views from the data flow editor. If you insert a DataSource, the source system is automatically inserted with it as a separate node in the data flow.
Insert persistent objects as nodes	You have the following options: • Drag & drop one or more objects from the Project Explorer • *Add Object* in the context menu for the grid In the popup, you can select one object or multiple objects (with the CTRL button). If you insert a CompositeProvider, its components, together with any SAP HANA views, are automatically inserted with it as separate nodes. If the SAP HANA system is linked with the BW project, you can open the SAP HANA views from the data flow editor. If you insert a DataSource, the source system is automatically inserted with it as a separate node in the data flow.
Copy and insert nodes	1. Select one or more nodes and choose *Copy* from the context menu. 2. Position the mouse on the line in the grid where you want to insert the nodes you are copying and choose *Paste* from the context menu. The system retains the layout of the copied nodes when inserting them. Copies of persistent objects are inserted into the dataflow object as non-persistent objects.
Move nodes in the grid	Select one or more nodes and move these with drag & drop.

Function	Explanations
Remove nodes from the dataflow object	You have the following options: • Select one or more nodes and remove these from the dataflow object by choosing *Remove Node* from the context menu or by pressing the Remove button. • For a single object, you can also use the  (*Remove from data flow*) function in the context pad. When removing nodes from the data flow object, no objects are deleted from the SAP BW/4HANA Repository.

Connections

Function	Explanations
Create connections between nodes: Non-persistent transformations, data transfer processes and InfoPackages, together with relationships between objects and their constituent parts	You have the following options: • Drag and drop from the source object (onto → (*Drag to create connections*) in the context pad, or from the gray area on the upper edge of the node onto the target object. • Single click on → (*Drag to create connections*) in the context pad of the source object and then single click on the target object. • To create multiple connections to target and/or source objects, choose *New Connection* from the context menu on a node. The following applies to connections between the source object and target object: A non-persistent transformation is created as the first connection. All other connections are created as non-persistent data transfer processes. Connections from a object to its components can only be dragged between non-persistent objects. This is not suitable for persistent objects because the components of the CompositeProvider, for example, or the source system of a DataSource can be specified, in the object maintenance screen, as part of the meta definition of the object.

Function	Explanations
Highlight connections from one node to another nodes in the display	To be able to recognize all connections from one node to other nodes in the data flow at a glance, you can highlight them. You have the following options: • *Highlight Relationships* in the context menu for the node • Single click on  (*Highlight relationships*) in the context pad of the node When you highlight the connections of one node to other nodes, then this node and its connections to other nodes are displayed in color. The icons for the different connection types are displayed on the relevant nodes for transformations and data transfer processes. The number of connections is shown next to the icon. Hover with the mouse pointer to see how many connections are persistent and non-persistent.

Function	Explanations

Create persistent transformations and data transfer processes from non-persistent connections	Prerequisite: The source nodes and target nodes are persistent objects and the corresponding non-persistent connections are available.
	
	You have the following options:
	
	• Double-click on the transformation or DTP icon in the target node
	If there is already more than a data transfer process between the source object and target object, you can select the required data transfer process.
	• The context menu for the node contains the following entries:
	  • *Transformations* > *Create <Name of the Transformation>*
	  • *DTPs* > *Create <Name of the DTP>*
	• Select a node and choose *Highlight Relationships* from the node context menu. The icons for the connections are displayed on the relevant nodes.
	Double-click on the transformation icon to call the maintenance screen for transformations, where you can create transformations.
	If you double-click on the data transfer process icon, a dialog opens, where you have the following options:
	  • *Insert Existing DTPs*
	  If you select this option, the existing data transfer processes are inserted between the source and the target. If there is no data transfer process between the source and the target, you can create a data transfer process. If there are no transformation between the source and the target yet, the system creates a default transformation.
	  • *Insert Existing DTPs and Create New DTP*
	  If you select this option, the existing data transfer processes between the source and the target. The screen for creating an additional data transfer process appears.
	  • *Display Existing DTPs in Transient Editor*
	  If you choose this option, you can display the existing data transfer processes before you create a new data transfer process.
	If the objects already exist, you can call the object maintenance screen by double-clicking on them.
	• *Create DTP* in the context menu for the node

Function	Explanations
	• If the selected node should serve as a source: Select a target node in the *Targets* tab page. • If the selected node should serve as a destination: Select a source node in the *Sources* tab page. The selection window described above appears.
Display connection details	*Show Connections* in the context menu for the node On the *Connections* tab page in the *Properties* view , all of the selected node's incoming and outgoing connections are listed, together with information about the source/target.
Remove connection	*Remove Connection* in the context menu for the node The system displays all of the node's incoming and outgoing connections that can be removed from the data flow object.
Display incoming connections for a node	Entry *Incoming: <connection>* in the context menu for the node. Under *Display <Name of the connection>*, the system displays the incoming persistent connections. Under *Create <Name of the connection>*, you can create a persistent connection from a non-persistent one.

Data flows

Function	Explanations
Insert data flow of an existing data flow object	*Insert Data Flow* in the context menu for the grid When you insert a data flow, all persistent objects in the data flow are added to the data flow object as persistent objects if they are not there already.
Insert data flow of an existing data flow object as a template	You have the following options: • *Insert Data Flow As Template* in the context menu for the grid • Drag & drop one or more data flows to the grid from the Project Explorer If you insert a data flow as a template, all inserted nodes are inserted as non-persistent objects.

Documentation

Function	Explanations
Display or create documentation	You have the following options: • Documentation for a node: *Show Documentation* in the context menu for the node • Documentation for the data flow object: Open the *Properties* view and select an empty cell in the data flow object's grid in the editor screen for the data flow object.

Miscellaneous

Function	Explanations
Open the editor for persistent object	You have the following options: • Double-click on the node • *Open Editor* in the context menu for the node The system opens the object in the editor.
Display transient data flow for a persistent object	*Explore Data Flow* > *Explore Data Flow Downwards* or *Explore Data Flow* > *Explore Data Flow Upwards* in the context menu for the node The system opens an editor to display the transient data flow. The system displays the complete downward or upward data flow for the selected node.
Expand data flow for a persistent object	You have the following options: • *Explore Data Flow* in the context menu for the node • Single click on one of the expand symbols in the node's context pad For the object, you can expand the data flow upward or downward by a single level or completely.
Call InfoProvider administration	For InfoProvider that have an administration UI (for example DataStore-Objekt (advanced) or InfoObject): *Manage* in the context menu for the node

Function	Explanations
Change grid	*Insert* in the context menu for the grid You can insert rows, columns and cells into the grid: • *Insert Row* (top) • *Insert Column* (left) • *Insert Cell Left* • *Insert Cell Right*
Align nodes in the grid	Select one or more nodes in the data flow and choose *Alignment* from the context menu. Various alignment functions are available here, for example aligning the node left or right.
Print data flow object	*Print* in the context menu for the grid or for a node You can make various print settings, such as the size and the position.
Export data flow object as image file	*Export Diagram* in the context menu for the grid or for a node You can make various export settings, such as the size and the image format.

Related Information

Creating Documentation [page 208]

4.2.2 Creating Documentation

You create the documentation for the data flow object and its nodes in the context of the data flow. This means that the documentation is not part of the SAP BW/4HANA objects. The purpose of this documentation is more to provide a comment on the data flow and its nodes. You could do this for example if you have a data flow object with non-persistent objects and want to provide a colleague with a basic system architecture so that s/he can take over the technical implementation.

Procedure

1. Create documentation for the data flow itself and for the nodes of the data flow object:
   - To create the documentation for the data flow object, open the *Properties* view and select an empty cell in the data flow object's grid in the data flow object editor.

- To create the documentation for the node of the data flow object, choose *Show Documentation* from the context menu for the node in the data flow object editor.

In the *Properties* view, the *Documentation* tab page displays the interface for maintaining documentation.

2. Under *Documentation Type*, select the editor that you want to create the documentation with, and enter the text in the editor:
   - TEXT, to create a pure text file
   - HTML, to make use of enhanced editing and structuring options
3. For non-persistent objects: You can change the technical name and description of the object. The changes are taken over directly in all places where the node is displayed, for example in the node in the data flow editor or in the *Outline* view. The system checks whether the entries for the technical names comply with the naming convention for SAP BW/4HANA objects.

4.2.3 Saving Data Flow Documentation as an HTML File

You can display the data flow documentation for an entire data flow in HTML format as a preview, and save the documentation as an HTML file on a computer, or print it. This allows you to keep a record of the current modeling status and have a good basis for creating documentation for your BW project, or auditing it.

Context

The data flow documentation contains the following information:

- Header data: Date on which the HTML page was created, and the system that contains the data flow
- Overview table for the objects of the data flow. The object names are linked, and clicking these links displays information about the objects.
- For each active object in a data flow:
  - If available: Documentation created in the data flow editor for the data flow and the individual objects in the data flow
  - Metadata description from the Metadata Repository:
    - Table containing a description of the object containing the technical name, descriptions (short and long), object version, the user who last changed the object, the last change, and the logical system name
    - If available: Links to BW documents about the object
    - Table containing an overview of the metadata for the object, for example contained objects such as InfoObjects or object-specific details
    - For transformations:
      - Table showing an overview of field mapping
      - For each conversion, formula, or routine: List of fields used in the field mapping overview
      - Field details and rule details (can be displayed or hidden)
    - For data transfer processes:
      - List of fields used in the filter with information on the filter conditions

- List of key fields for error stack
- Tables containing overviews of objects in the environment, such as required objects, objects from which data was received, or objects to which data was passed.

Procedure

Calling the Data Flow Documentation as a Preview

Choose  (*Show Documentation*) in the toolbar in the data flow editor.

The system displays the HTML documentation for the data flow as a preview in a window.

> ⓘ Note
>
> You can only call the function if there is an active version of the data flow object, and the object version currently displayed in the editor is the active version. If you created a new data flow for example and saved it but not activated it yet, you cannot call the function. If there is already an active version, but you have unsaved changes in the editor in the editor, or If a revised version is displayed in the editor, the system displays the documentation for the active version.

Saving the Documentation from the Preview as a File

Choose  (*Save*) in the toolbar in the preview in order to save the data flow documentation as an HTML file on a computer. This saves the documentation as currently displayed in the preview, with the information either shown or hidden. If you open the HTML file, it is displayed in the browser. As in the preview, navigation options are available in the HTML file.

Printing the Documentation from the Preview

Choose (*Print*) in the toolbar in the preview in order to print the data flow documentation or to save it is a PDF file using a PDF print driver.

Related Information

Creating Documentation [page 208]

4.2.4 Copying a Data Flow

You can copy a data flow together with its objects. Using the simply copy function (*Copy*), you copy the data flow 1:1, and the copy contains the same objects as the original data flow. With the enhanced copy function, you can create multiple data flows and specify changes in comparison with the copied data flow. By choosing the copy mode, you can decide for example to swap objects in the copy with other objects, to create new objects or not to take over objects to the copied data flow.

Context

The enhanced data flow copy function supports various scenarios:

- You can use an existing data flow as a "master template" for other data flows.
- You can take over an existing data flow from SAP BW/4HANA Content, copy it and then modify the copy according to your requirements.
- You can copy an existing data flow for **one** source system for the other source systems you need.

When defining and performing a data flow copy, there is a wizard to help you. Here you select the objects that you want to take over to the target data flow and set the copy mode for the individual objects (for example by creating or using another existing object). You then assign the original objects to the target objects of the copy and - depending on the copy mode you have chosen - specify a technical name and a description for new target objects or select an existing object as the target object. Once you have specified what form the copy/copies should take, you finish the wizard in order to start the copy process.

> ⓘ Note
>
> If you want to use the enhanced data flow copy, the standard transport system must be deactivated.

Procedure

1. You are in the BW Modeling tools. Here you select *Copy Enhanced* in the context menu of the required data flow in the *Project Explorer*.
2. In the first wizard step, specify how many copies you want to create, and then choose *Next*.

   > → Tip
   >
   > If you want to create a simple 1:1 copy, we recommend using the *Copy* function in the context menu of the data flow that you plan to copy.
3. In the second step, the system displays the object types present in the original data flow below and lists the objects of the data flow below, with the copy mode and the assignment to the target objects in the target data flows.
4. Set the copy mode for the objects. To do this, highlight the row of the object in question and select the required copy mode under *Change Copy Type* in the toolbar. Alternatively, you can select the copy mode from the context menu of the row.

   Four copy modes are supported:

Copy Mode	Description
Original	The data flow copy contains the same object as the original data flow.
Different	In the data flow copy, the object is replaced with a different existing object.

Copy Mode	Description
Do not copy	The object is no longer contained in the data flow copy.
Create	In the data flow copy, the object is replaced by a newly created object. The new object is created as a copy of the original object during the copy process.

Which copy modes are available for a given object depends on the object type and on its dependencies to other objects. For the data flow object itself for example, only copy mode *Create* is available. For source systems, only *different* is available, meaning that a different source system can be used than in the original data flow. For transformations for example, the copy mode depends on the copy modes of the incoming and outgoing objects. If the incoming object in the copy is newly created for example, a new transformation must be created during the copy.

5. If necessary, edit the objects with copy mode *Create* or *Different*. To do this, select the row of the object in question and choose *Edit* in the toolbar. Alternatively, you can double-click the row.
   a. Create: In the dialog box, you can modify the proposed technical names and descriptions of the target objects.
   b. Different: In the dialog box, you can select a different object.

> ⓘ Note
>
> If objects such as transformations or data transfer processes are newly created, the technical name and description are created during the copy process. It is therefore not possible to set them manually in this step.

6. The *Status* column shows you whether or not the assignments of an original to the target objects are correct. If the status is incorrect, correct the assignments. Check for example whether the names of the target objects adhere to the naming conventions for the object type, or whether you have actually selected a different object if using copy mode *Different*.
7. Once you have made all the settings for the copies, and all objects have the status green, choose *Finish* to start the copy process.
8. The system displays the *BW Job View*. This contains an overview of the copy processes. Refresh the display until your current data flow process is displayed.
9. Select the data flow process in order to display the associated job details and logs.

Next Steps

In the InfoArea tree of your original data flow, choose *Refresh* so that you can see the copied, new data flows in the Project Explorer.

4.2.5 Using a Transient Data Flow for a Persistent Object

From a persistent SAP BW/4HANA object, you can call the editor for a transient data flow. In the editor, you can show the data flow of the object and thus better understand the relationships and dependencies between the various objects in the data flow.

1. Call the Editor for a transient data flow. You have the following options:
   1. In the *Project Explorer* view, you can choose *Explore Data Flow* in the context menu of an object.
   2. In the editor of a data flow object, you can open the object context menu and choose *Explore Data Flow* > *Explore In New Data Flow*.

   The system opens the editor for the transient data flow and displays the object in question as a node.
2. The editor for a transient data flow has the same structure and functions as the Data Flow Object Editor. In particular, you can do the following here:
   - By starting from one object, you can using the transient editor to model further objects in the data flow and create connections between the objects in the data flow.

     > **Note**
     >
     > However, please note that non-persistent objects are discarded when you quit the editor, unless you save the data flow as a data flow object (see below).

   - By starting from one object, you can display and analyze the data flow.
     You can do this by opening the context menu or the context pad from the selected object.
     - Show the targets or sources of the object one level higher or lower in the data flow
     - Show the entire upward or downward data flow for the object

     Once you have shown data flows for various objects and on various levels, the alignment of nodes and connections in the grid might not be as tidy as before. In this case, you can use the *Auto Layout* function in the editor toolbar to automatically arrange the nodes and connections and make the display tidier.
   - You can save the transient data flow displayed in the editor as a data flow object.
     Open the Eclipse menu and choose *File* > *Save As*. In the popup that appears now, enter an InfoArea, a technical name and a description for the data flow object.

Related Information

Working with the Data Flow Object Editor [page 198]
Functions in the Data Flow Object [page 201]

4.2.6 Tips and Tricks for Working with Data Flows

Easily Connecting Objects

Creating a Template for a Data Flow

4.2.7 Recommendations for Working with Large Data Flows

Various functions are available to make it easier to work with large data flows.

The following views provide helpful information and functions:

- The *Outline* view provides you with an overview of the nodes in the data flow and allows you to find nodes in the graphical editor quickly and easily.
- The *Properties* view provides you with a list of all incoming and outgoing connections to a node selected in the editor.
- The *Miniature* view shows you where the section displayed in the editor is situated in the data flow.

The following functions in the graphical editor are particularly helpful:

- *Highlight Relationships* in the context menu for a node. This allows you to recognize all incoming and outgoing connections for a node with other nodes in the displayed data flow by highlighting them.
- *New Connection* in the context menu for a node. This provides you with a popup containing a list of all available source and target nodes.
- Various functions in the editor toolbar help you to have a clearer overview of the data flow in the editor, for example by compressing the view, zooming, or automatically adjusting the layout.

Related Information

Working with Data Flows [page 197]

4.3 Creating an Open ODS View

Open ODS views allow the flexible integration of external data sources into SAP BW/4HANA, without the need to create InfoObjects. In BW modeling tools, you can create simple Open ODS views and enhance them in stages for closer data integration.

Creating a Simple Open ODS View

1. You are in the BW Modeling tools. In the *Project Explorer* view, choose *New* > *Open ODS View* in the context menu of your BW project or the context menu of an InfoArea or the semantic folder *Open ODS View*. There is a wizard to guide you through the process of creating the open ODS view.
   More information: Creating an Open ODS View Using the Wizard (Defining Semantics and Source) [page 216]
2. In the Open ODS view maintenance screen, you specify the structure of the Open ODS view. To do this, sort the fields of the source object into field categories of the Open ODS view. The selection of categories available for assigning fields depends on the semantics of the Open ODS view. Field categories for facts include characteristics, key figures, currencies or units.
   The properties of the Open ODS view fields are set by either the source of the Open ODS view field or a system default. The properties are different for fields of different categories. They describe, for example, the aggregation behavior for key figures. For characteristics, for example, the properties specify whether the characteristics are authorization-relevant.
   More information:
   Specifying the Structure of the Open ODS View [page 224]
   Displaying Initial Properties for Open ODS View Fields [page 225]
3. You can now check, save and activate the Open ODS view. This means the view is now available for use and you can virtually use the data source of the Open ODS view, for example, in queries. In further steps you can also flexibly enhance the Open ODS view.
   1. Check the Open ODS view ( ).
      The system displays the check log in the problems view in BW modeling tools.
   2. [Optional:] Check the query behavior of the open ODS view in the *BW Reporting Preview* view.
      More information: BW Reporting Preview View [page 15]
   3. Save ( ) and activate ( ) the Open ODS view.

Additional Editing for Open ODS Views

1. If you perform further editing on an Open ODS view, you can specify various properties and combine the data source of the Open ODS view with SAP BW/4HANA models (InfoObjects).
   - You can adjust and change the properties of the Open ODS view fields. In particular, you can add associations of other Open ODS views and/or InfoObjects to fields:
     - Using associations on fields of type "characteristic", you can link an Open ODS view to master data and texts of other Open ODS views or InfoObjects. This allows you to inherit properties from

the associated object, assign additional key fields (compounding) and use texts and navigation attributes of the associated objects in queries on the Open ODS view.

- In the case of fields with type "key figure", you can add associations to InfoObjects. This is particularly relevant if you want to reuse formulas or variables (defined on the associated InfoObject) in queries.

- As with an InfoObject, the Open ODS view can combine language-independent semantics for the attribute keys with language-dependent semantics for texts. You can add extra semantics for an Open ODS view that you initially created with one of the two other semantics (texts or attributes).
- You can swap the source object in the Open ODS view.

More information:
Changing Properties for Fields of Type "Characteristic" [page 228]
Changing Properties for Fields of Type "Key Figure" [page 231]
Adding Texts or Attributes [page 232]
Replacing the Source of an Open ODS View [page 233]

2. After you have finished the additional editing, check, save and activate the Open ODS view.

4.3.1 Creating an Open ODS View Using the Wizard (Defining Semantics and Source)

There is a wizard to guide you through the process of creating an open ODS view. In the wizard, you make general entries for the technical name and description, InfoArea and favorites assignments, and specify whether a proposal should be made for the assignment of the fields in the source object to field categories of the Open ODS view when this is created. During the creation process in the wizard, you select the semantics and the source of the Open ODS view. You also have the option of creating an Open ODS view as a copy of an existing Open ODS view.

Prerequisites

- For transformations as sources of Open ODS views:
  You can use a transformation as the source for the Open ODS view provided that you have created the corresponding data flow:
- For DataStore objects as sources of Open ODS views:
  DataStore objects for which all reporting takes place on the active table are supported.

Procedure

In the *Project Explorer* view, you have chosen *New* > *Open ODS View* in the context menu of your BW project or the context menu of an InfoArea or the semantic folder You are in the wizard for creating Open ODS views.

1. In the first wizard step *Open ODS View*, enter some basic information.

a. Select an InfoArea.

   The InfoArea that you selected the last time you opened the dialog box is displayed as the default entry.

   > → Tip
   >
   > You can access the content assist function by pressing `Ctrl + Space`.

b. To access the Open ODS faster next time you are here, we recommend setting the *Add to Favorites* flag.
c. Enter a technical name for the Open ODS view under *Name*. A maximum of 12 characters can be entered for the name.
d. Enter a description for the Open ODS view under *Description*.
e. The *Create with proposal* flag is selected by default. This means that the Open ODS view is created with automatic assignment of the source fields to the field categories of the Open ODS view (characteristic or key figure for example). If you deselect this, the system does not generate a proposal during the creation process. In this case, you set the assignment in the editor of the Open ODS view.
f. Under *Semantics*, specify whether you want to create an Open ODS view for facts, master data or texts.
g. Choose *Next*.

2. In the *Select Source Type* wizard step, select which type of source object should be used as the basis for the Open ODS view.

   For the source types *Database Table or View*, *Database Table or View via HANA MDC*, *Virtual table using SAP HANA Smart Data Access*, and *Big Data*, specify whether an existing SAP HANA/Big Data source system, or a new source system, should be used to access the source object of the Open ODS view.

3. Choose *Next*.

   The following wizard steps depend on which source type you have selected and - for specific source types (see step 2) - whether you use an existing source system or create a new one.

   More information:

   - Selecting the Source for Type Database Table or View [page 218]
   - Selecting the Source for Type Database Table or View via HANA MDC [page 219]
   - Selecting the Source for Type Virtual Table Using SAP HANA Smart Data Access [page 220]
   - Selecting a Source for the Big Data Type [page 222]
   - Selecting the Source for Type DataSource (BW) [page 223]
   - Selecting the Source for Type Transformation [page 223]
   - Selecting the Source for Type DataStore Object (advanced) [page 224]

Results

Once you have completed all wizard steps, the Open ODS View editor opens.

4.3.1.1 Selecting the Source for Type Database Table or View

In these wizard steps, you select a database table or a view from a local SAP HANA database schema as the source of the Open ODS view. To do this, you use an existing source system as the connection to the SAP HANA, or create a new SAP HANA source system.

Use Existing Source System

In the previous wizard step, you selected *Existing Source System*.

1. In the wizard step *Select a Source System*, enter a search term and select the source system from the list of search results.
2. Choose *Next*.
3. In the wizard step *Select a Database Object*, enter a search term and select a table or a view from the list of search results.
4. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
5. To create the Open ODS view, choose *Finish*.

Create New Source System

In the previous wizard step, you selected *Owner / Schema*.

1. In the wizard step *Select a DB Object Schema*, enter a search term and select the schema (owner) from the list of search results.
2. Choose *Next*.
3. In the wizard step *Select a DB Object*, enter a search term and select a table or a view from the list of search results.
4. Choose *Next*.
5. In the wizard step *Select a Source System* under *Create new*, enter a name for the source system you want to create.

   > ⓘ Note
   >
   > If source systems already exist for the schema (owner) that you previously selected, the system proposes the use of an existing source system and displays a list of suitable source systems. If you decide against creating a new source system, you can select an existing source system here instead. If you navigate back in the wizard and change the selection of the schema, the proposal for existing source systems changes too.

6. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
7. To create the Open ODS view, choose *Finish*.
   The Open ODS View Editor appears.

The system creates a SAP HANA source system with connection type *Local SAP HANA Database Schema*.

Related Information

Connection Types for the SAP HANA Source System [page 143]

4.3.1.2 Selecting the Source for Type Database Table or View via HANA MDC

In these wizard steps, you select a database table or a view from a SAP HANA tenant database schema as the source of the Open ODS view. To do this, you use an existing source system as the connection to the SAP HANA, or create a new SAP HANA source system.

Use Existing Source System

In the previous wizard step, you selected *Existing Source System*.

1. In the wizard step *Select a Source System*, enter a search term and select the source system from the list of search results.
2. Choose *Next*.
3. In the wizard step *Select a Database Object*, enter a search term and select a table or a view from the list of search results.
4. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
5. To create the Open ODS view, choose *Finish*.

Create New Source System

In the previous wizard step, you selected *Remote Database*.

1. In the wizard step *Select a Remote Database*, enter a search term and select the database from the list of search results.
2. Choose *Next*.
3. In the wizard step *Select a DB Object Schema*, enter a search term and select the schema (owner) from the list of search results.
4. In the wizard step *Select a DB Object*, enter a search term and select a table or a view from the list of search results.
5. Choose *Next*.
6. In the wizard step *Select a Source System* under *Create new*, enter a name for the source system you want to create.

> **Note**
>
> If for the schema (owner) that you selected source systems already exist, the system proposes using an existing source system and displays a list of suitable source systems. If you decide against creating a new source system, you can select an existing source system here instead. If you navigate back in the wizard and change the selection of the schema, the proposal for existing source systems changes too.

7. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
8. To create the Open ODS view, choose *Finish*.
   The Open ODS View Editor appears.
   The system creates a SAP HANA source system with connection type *SAP HANA TenantDatabase Schema*.

4.3.1.3 Selecting the Source for Type Virtual Table Using SAP HANA Smart Data Access

In these wizard steps, you select a source object from an external source connected by SAP HANA Smart Data Access as the source of the Open ODS view. To do this, you use an existing source system as the connection to the SAP HANA, or create a new SAP HANA source system.

Use Existing Source System

In the previous wizard step, you selected *Existing Source System*.

1. In the wizard step *Select a Source System*, enter a search term and select the source system from the list of search results.

   > **Note**
   >
   > Note that source systems for remote sources with the adapter *Spark SQL DESTINATION* are not displayed here, because a dedicated source type (Big Data) is available for these systems.

2. Choose *Next*.
3. In the wizard step *Select a Database Object*, enter a search term and select a table or a view as the source object from the list of search results.

   > **Note**
   >
   > For sources with a logical structure (folder), the system displays the path suffix next to the object name in the results list.

4. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
5. To create the Open ODS view, choose *Finish*.

Create New Source System

In the previous wizard step, you selected *Remote Source*.

1. In the wizard step *Select a Remote Source*, enter a search term and select the source system from the list of search results.

   > ### Note
   >
   > Note that a dedicated source type (Big Data) is available for remote sources with the adapter *Spark SQL DESTINATION*. Therefore remote sources with this adapter are not available for source type *Virtual Tables via SAP HANA Smart Data Access*.

2. Choose *Next*.
3. For sources with schemas:
   1. In the wizard step *Select a Database Object*, enter a search term and select a schema from the list of search results.
   2. Choose *Next*.
4. In the wizard step *Select a Database Object*, enter a search term and select a table or a view as the source object from the list of search results.

   > ### Note
   >
   > For sources with a logical structure (folder), the system displays the path to the source object next to the object name in the results list.

5. In the wizard step *Select a Source System* under *Create new*, enter a name for the source system you want to create.

   > ### Note
   >
   > If for the schema (owner) or for the path that you previously selected, source systems already exist, the system proposes the use of an existing source system and displays a list of suitable source systems. If you decide against creating a new source system, you can select an existing source system here instead. If you navigate back in the wizard and change the selection of the schema or path, the proposal for existing source systems changes too.

6. For sources with a logical structure (folder): Under *Path Prefix* choose from which part of the path of the source object is to be specified as a prefix in the source system setup.

   > ### Note
   >
   > The part of the path that is not specified as a prefix here becomes the suffix part of DataSource definition.

7. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
8. To create the Open ODS view, choose *Finish*.
   The Open ODS View Editor appears.
   The system creates a SAP HANA source system with connection type *SAP HANA Smart Data Access*.

Related Information

Connection Types for the SAP HANA Source System [page 143]
Path Prefix and Suffix [page 145]

4.3.1.4 Selecting a Source for the Big Data Type

In these wizard steps, you select a source object from a Big Data source (Hadoop) as the source of the Open ODS view. To do this, use an existing source system as the connection to the Big Data source, or create a new Big Data source system.

Using an Existing Source System

In the previous wizard step, you selected *Existing Source System*.

1. In the wizard step *Select a Source System*, enter a search term and select the source system from the list of search results.
2. Choose *Next*.
3. In the wizard step *Select a Database Object*, enter a search term and select a source object from the list of search results.
4. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
5. To create the Open ODS view, choose *Finish*.

Create New Source System

In the previous wizard step, you selected *Remote Source*.

1. In the wizard step *Select a Remote Source*, enter a search term and select the source system from the list of search results.
2. Choose *Next*.
3. In the wizard step *Select a Database Object*, enter a search term and select a schema from the list of search results.
4. In the wizard step *Select a Database Object*, enter a search term and select a source object from the list of search results.
5. In the wizard step *Select a Source System* under *Create new*, enter a name for the source system you want to create.

> **Note**
>
> If source systems already exist for the schema that you previously selected, the system proposes the use of an existing source system and displays a list of suitable source systems. If you decide against creating a new source system, you can select an existing source system here instead. If you navigate

> back in the wizard and change the selection of the schema, the proposal for existing source systems changes too.

6. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
7. To create the Open ODS view, choose *Finish*.
   The Open ODS View Editor appears.
   The system creates a Big Data source system.

4.3.1.5 Selecting the Source for Type DataSource (BW)

In this wizard step, you select a DataSource from the SAP BW/4HANA system as the source of the Open ODS view.

Procedure

1. In the wizard step *Select a DataSource*, enter a search term and select the DataSource from the list of search results.
   The DataSource information is displayed in the list of search results using the following convention: [<technical name of DataSource>] <description of DataSource> (name of source system).
2. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
3. To create the Open ODS view, choose *Finish*.
   The Open ODS View Editor appears.

4.3.1.6 Selecting the Source for Type Transformation

In this wizard step, you select a transformation as the source for the Open ODS view by specifying both ends of the transformation (DataSource as source of the transformation and InfoSource as target).

Procedure

1. In the wizard step *Select a Transformation*, under *Object Type*, specify whether you want to specify the source of the transformation (the DataSource) first or the target (the InfoSource).
2. Enter a search term and select the object from the list of search results.
   The system only displays DataSources that have an InfoSource as their target, and InfoSources that have a DataSource as their source.
3. To define the other end of the transformation, choose *Next*.

The wizard now shows the displayed end of the transformation. Under *Object Type*, the object type for the other end of the transformation is already displayed.

4. Enter a search term and select the object from the list of search results.

   In the list of search results for the other end of the transformation, the system only displays suitable objects, i.e. Objects that have the other end of the transformation (as just defined) as their source or target.

5. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
6. To create the Open ODS view, choose *Finish*.

   The Open ODS View Editor appears.

4.3.1.7 Selecting the Source for Type DataStore Object (advanced)

In this wizard, you select a DataStore object (advanced) as the source of the Open ODS view.

Procedure

1. In the wizard step *Select a DataStore Object (advanced)*, enter a search term and select the DataStore object (advanced) from the list of search results.
2. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
3. To create the Open ODS view, choose *Finish*.

   The Open ODS View Editor appears.

4.3.2 Specifying the Structure of the Open ODS View

In the Open ODS view editor, you can specify the structure of the Open ODS view on the semantics tab, by assigning the fields of the source object to field categories (example: for facts, assign to the categories characteristic key, characteristic, key figure and others).

Procedure

1. Go to the Semantics tab.

   When creating the Open ODS vew, you use the *Create with Proposal* flag to specify whether the Open ODS view should be created with automatic assignment of the source fields to the field categories of the Open ODS view. In the default settings, automatic assignment is carried out. In this case, you see the automatic

assignment of source fields to field categories of the Open ODS view in the middle screen area. A proposal is generated based on the source type.

2. You can change the system proposal if required. If you created the Open ODS view without automatic assignment, define the structure of the Open ODS view. Various options are available here.

Option	Description
Drag and Drop	You can drag source fields into the field categories of the Open ODS view or move Open ODS view fields from one field category to another.
Add **in the context menu of a source field**	In the *Add Source Fields* dialog box , choose *Field Type* and select the category that you want the field to be assigned to. Confirm your selection by pressing *OK*.
Move **in the context menu of an Open ODS view field.**	In the *Move Fields* dialog box, choose *Field Type* and select the category that you want to assign the field to. Confirm your selection by pressing *OK*.
Remove **in the context menu of an Open ODS view field.**	The field is removed from the structure of the Open ODS view and is colored gray in the list of source fields, if it is not assigned to any field categories.

> ⓘ Note
>
> - Unassigned source fields are colored gray (in the left screen area of the tab).
> - You can move a source field multiple times in the structure of the Open ODS view, in order to assign it to different field categories. Source fields assigned multiple times are colored blue (in the left screen area of the tab).
> - If multiple key fields exist for attributes or texts, drag the key field that defines or represents the Open ODS view to field category *Representative Key Field*.

Results

In the middle screen area of the tab, you see the structure of the Open ODS view.

The names of the Open ODS view fields are copied from the source fields. The names of the view fields are limited to 12 characters. The description of the view fields is copied from the description of the source fields. If the source does not have a description, the field names are used for the description of the view fields.

You can now display the properties of the Open ODS view fields and change the predefined properties.

Related Information

Displaying Initial Properties for Open ODS View Fields [page 225]
Changing Properties for Fields of Type "Characteristic" [page 228]
Changing Properties for Fields of Type "Key Figure" [page 231]

4.3.3 Displaying Initial Properties for Open ODS View Fields

In the right screen area of the semantics tab in the Open ODS View Editor, you can view and modify the properties of the Open ODS view fields. A field can have different properties, depending on the field type. The

field properties are initially inherited from the source of the Open ODS view (whenever possible). A default is used for any properties that cannot be identified in the source.

The right screen area of the semantics tab shows the properties of the Open ODS view field, which is selected in the structure of the Open ODS view in the middle screen area.

General Field Properties

All types of fields have the following properties, which are displayed under *General*:

Property	Description
Name	The field name is read from the source field. Lowercase letters in the source field are converted to uppercase letters, and the name length is limited to 12 letters.
Description	The description is read from the source field. If the source field does not have a description, the name of the source field is used as the description of the Open ODS view field.

InfoObject Name

A further property of an Open ODS view field is the InfoObject name. This is the name for the Open ODS view field used across the system, for example in the InfoProvider display of the Open ODS view or in a query on the Open ODS view. It is determined initially from the ODP name and the name of the Open ODS view field and is displayed in *Association*, under *Usage of System-Wide Unique Name*.

Further Properties for Fields of Type Characteristic

Fields of type "characteristic" have the following additional properties:

Property		Description
Association		Using associations on fields of type "characteristic", you can link an Open ODS view to master data and texts of other Open ODS views or InfoObjects. This makes it possible to inherit properties - such as the InfoObject name, the authorization relevance or reporting properties of the associated object - and use texts and navigation attributes of the associated objects in the InfoProvider of the Open ODS view. Initially there are no associations.
Characteristic-Specific Properties	*Authorization Relevance*	This property specifies whether analysis authorizations are checked at query runtime and which authorizations are checked (if the field is authorization-relevant). Initially the Open ODS view field is not authorization-relevant.
	Referential integrity (*I confirm the Referential Integrity*)	This property is relevant only if the field of the Open ODS view has an association to an InfoObject. In this case, it defines how the data is handled at query runtime with respect to the referential integrity. In the initial setting, this flag is not selected.
Reporting Properties	*Display*	This property specifies whether individual characteristic values in the query are displayed and how they are displayed. Initially the characteristic values are displayed with their keys.
	Query Execution Filter	This property specifies how filter values are selected for query execution: Initially only posted values are displayed.

Properties for Fields of Type "Key Figure"

The fields of type key figure have the following properties:

Property		Description
Association		Using associations on fields of type "key figure", you can link an Open ODS view with InfoObjects. This allows you for example to use formulas and variables (defined on the associated InfoObject) in queries on the Open ODS view and inherit properties (such as the InfoObject name, aggregation or reporting properties) from the associated InfoObject. Initially there are no associations.
Key Figure-Specific Properties	*Currency / Unit Element*	This property specifies which field in the Open ODS view is used as the currency field or unit field for the key figure. Initially a unit field or currency field is not assigned.
	Aggregation Behavior	This property specifies which rules are used for key figure aggregation. Initially the field has the aggregation behavior "summation".
Reporting Properties	*Decimal Places*	This property specifies how many decimal places are used for the field in the query. Initially two decimal places are set.
	Digits to Display	This property specifies the scaling factor that is used to display key figure values in the query. Initially scaling factor 1 is set.

4.3.4 Changing Properties for Fields of Type "Characteristic"

In the right screen area of the semantics tab in the Open ODS View Editor, you can view and modify the properties of the Open ODS view fields. You can change characteristic properties for fields of an Open ODS view by changing the selections/entries in the properties maintenance screen (overwrite local settings). Alternatively, you can specify an association for the field and this association can pass on various properties to the Open ODS view field.

Changing local field properties

To change a field property locally, proceed as follows:

- For field properties like the name of the view field or the association, change the entry or the selection directly.
- For field properties that display the  (*Toggle Default Value/Manual Entry*) pushbutton (such as the reporting properties), choose the pushbutton for the property and change the entry or selection.

> **Note**
>
> Even if you do not change the entry after pressing  (*Toggle Default Value/Manual Entry*), the system interprets the property as being specified locally and not as the default or inherited value.

> **Note**
>
> The authorization behaviour of an Open ODS view field depends on whether the *Authorization Relevance* property has been set locally or via an association.

Creating field associations and specifying field properties using associations

Using associations on fields of type "characteristic", you can link an Open ODS view to master data and texts of other Open ODS views or InfoObjects. This allows you to inherit properties from the associated object, assign additional key fields (compounding) and use texts and navigation attributes of the associated objects in queries on the Open ODS view.

1. To specify an association, choose the associated object type under *Association* (*Open ODS View* or *InfoObject*) and then select the required object.
   Creating the association for an Open ODS view field transfers the *Authorization Relevance* properties and the reporting properties (*Display*, *Query Execution Filter*) from the associated InfoObject to the Open ODS view field. You can also specify additional properties for the field.

   > **Note**
   >
   > The authorization behaviour of an Open ODS view field depends on whether the *Authorization Relevance* property has been set locally or via an association.

2. If you want the info object name for the Open ODS view field to be inherited from the associated object, choose *Direct Usage of Associated Object by Name*.
   If you choose *Direct Usage of Associated Object by Name*, and the associated object is an InfoObject, the InfoObject name of the Open ODS view field matches the name of the associated InfoObject. If you choose *Direct Usage of Associated Object by Name*, and the associated object is an Open ODS view, the InfoObject name of the Open ODS view field matches the name of the associated Open ODS view.
3. To specify navigation attributes from the associated object for the Open ODS view, choose *Navigation Attributes*.
   This pushbutton is available for an association to an InfoObject, provided that the InfoObject contains navigation attributes. This pushbutton is available for an association to an Open ODS view, provided that the Open ODS view contains characteristics.
   1. A dialog box opens. Select either navigation attributes of the associated InfoObject or select characteristics of the Open ODS view as navigation attributes of the Open ODS.
   2. If you have chosen *Usage of System-Wide Unique Name*, you can change the field names in the Open ODS view for the selected navigation attributes (under *Local Name*). The field names are used to derive the InfoObject names of the navigation attributes.

3. You can also change the description of a navigation attribute to be used in queries for an Open ODS view. Enter a description in the *Local Description* field.
4. To return to the Open ODS view maintenance screen, choose *OK*.

4. To specify a compounding to another field, choose *Compounding*.
   This pushbutton is available for an association to an InfoObject, provided that the InfoObject contains a different characteristic. This pushbutton is available for an association to an Open ODS view, provided that the associated Open ODS view contains the represented key field and at least one additional key field.
   1. A dialog box opens. Here you see the compounding parent InfoObjects of the associated InfoObject or the key fields of the associated Open ODS view under *Compounding from Association*.
   2. Under *Field Name* choose the Open ODS view field which you want compound to.
   3. Once you have specified the compounding, choose *OK* to return to the Open ODS view maintenance screen.

   > ⓘ Note
   >
   > If you have just specified a compounding for a field and this field does not have an association to an Open ODS view - but an association was specified in the Open ODS view of the compounding parent for the source field - then this association is applied when the Open ODS view is checked.

5. For associated InfoObjects: The *I confirm the Referential Integrity* field is a reminder to make sure that the required master data for all the characteristic values is available in the system. The query result is modified by the field in the following way:
   If you select the field, a check on referential integrity is made when the query is executed. This means that data is displayed only for those characteristics values where entries exist both in the external source and in the SID table of the associated info object.
   If you do not select the field, a check on referential integrity is not made when the query is executed. This means that data is also displayed for those characteristics values where entries exist only in the external source and no entries exist in the SID table of the associated info object.

   > ⓘ Note
   >
   > Note the effects of this field on the query result and on the master data table of the associated info object. If you do not select the field and the referential integrity is not confirmed, this can produce a different query result than in the referential integrity check and the master data table of the associated info object is given additional entries when the query is executed.

> ⓘ Note
>
> Properties specified using the association can be modified again on the property; choose  (*Toggle Default Value/Manual Entry*). You can also use this pushbutton to reset local changes.

Related Information

Authorization Behaviour for Fields of Open ODS Views [page 51]

4.3.5 Changing Properties for Fields of Type "Key Figure"

In the right screen area of the semantics tab in the Open ODS View Editor, you can view and modify the properties of the Open ODS view fields. You can change key figure properties for fields of an Open ODS view by changing the selections/entries in the properties maintenance screen (overwrite local settings). Alternatively, you can specify an association for the field and this association can pass on various properties to the Open ODS view field.

Changing local field properties

To change a field property locally, proceed as follows:

- For field properties like the name of the view field, change the entry or the selection directly.
- For field properties that display the  (*Toggle Default Value/Manual Entry*) pushbutton (such as the reporting properties), choose the pushbutton for the property and change the entry or selection.

> **Note**
>
> Even if you do not change the entry after pressing  (*Toggle Default Value/Manual Entry*), the system interprets the property as being specified locally and not as the default or inherited value.

> **Note**
>
> The currency field or unit field under *Currency/Unit Element* is always assigned locally. There is no default and no inheritance for this property. You can use Open ODS view fields (field types "currency" and "unit") when performing the assignment.

Creating field associations and specifying field properties using associations

Using associations on fields of type "key figure", you can link an Open ODS view with InfoObjects. This allows you, for example, to reuse formulas and variables (defined on the associated InfoObject) in queries on the Open ODS view and inherit properties from the associated object.

1. To specify an association to an InfoObject, choose *InfoObject* under *Association* and select the required object.
   Creating the association for an Open ODS view field transfers the *Aggregation Behavior* properties and the reporting properties (*Decimal Places*, *Digits to display*) from the associated InfoObject to the Open ODS view field.
   The info object name of the Open ODS view field is also inherited by the InfoObject and is displayed under *Direct Usage of Associated Object by Name*.
2. The *Currency/Unit Element* property must be consistent with the associated InfoObject. Make sure that specified currency field or unit field matches the InfoObject properties.

> **Note**
>
> Properties specified using the association can be modified again on the property; choose  (*Toggle Default Value/Manual Entry*). You can also use this pushbutton to reset local changes.
>
> Note that the field properties must be consistent with the associated InfoObject. If you overwrite properties inherited from the InfoObject, the changed entries must match the InfoObject properties.

4.3.6 Adding Texts or Attributes

To enable an Open ODS view to contain attributes as well as texts, you can add further semantics, apart from the semantics specified during creation, to the view. You can add texts to Open ODS views with attributes, and you can add attributes to Open ODS views with texts.

Context

For example, if, in the first step, you require a simple data model for which you want to use only texts about facts Open ODS views, you can add the attribute modeling during a later step. This means that the function also supports flexible modeling of Open ODS views.

Procedure

1. On the *General* tab under *Semantics* (for the semantics that you want to create), choose *New*.

   The wizard for creating semantics appears.
2. In the *Add Semantics* wizard step, under *Source Type*, select which type of source object should be used as the basis for the Open ODS view.
3. For the source types *Database Table or View* and *Virtual table using SAP HANA Smart Data Access*, specify whether an existing SAP HANA source system, or a new SAP HANA source system, should be used to access the source object of the Open ODS view.
4. Choose *Next*.

   The following wizard steps depend on which source type you have selected and - for the source types *Database Table or View* and *Virtual table using SAP HANA Smart Data Access* - whether you use an existing source system or create a new one.

   More information:

   - Selecting the Source for Type Database Table or View [page 218]
   - Selecting the Source for Type Virtual Table Using SAP HANA Smart Data Access [page 220]
   - Selecting the Source for Type DataSource (BW) [page 223]
   - Selecting the Source for Type Transformation [page 223]

- Selecting the Source for Type DataStore Object (advanced) [page 224]

5. Once you have completed all wizard steps, the Open ODS View editor opens. The Open ODS view now also displays the tab for the semantics that have been added. Specify the structure of the Open ODS view for the added semantics.

   > **Note**
   >
   > The system automatically performs the key field assignment where possible.

6. Check the key field assignment and make any required changes.
7. Save and activate the Open ODS view.

Results

The Open ODS view is ready for use.

You can now perform further modeling steps, for example changing field properties or adding associations.

4.3.7 Replacing the Source of an Open ODS View

On the *General* tab in the Open ODS View maintenance screen, you can replace the source of the Open ODS view with a different source. Provided that the Open ODS remains consistent after the source is swapped, objects that build on the Open ODS view - queries for example - can still be used.

Procedure

1. On the *General* tab, choose *Edit* under *Semantics*.

   The wizard for creating the source object appears.
2. In the *Change Source Object* wizard step, under *Source Type*, select which type of source object should be used as the basis for the Open ODS view.
3. For the source types *Database Table or View* and *Virtual table using SAP HANA Smart Data Access*, specify whether an existing SAP HANA source system, or a new SAP HANA source system, should be used to access the source object of the Open ODS view.
4. Choose *Next*.

   The following wizard steps depend on which source type you have selected and - for the source types *Database Table or View* and *Virtual table using SAP HANA Smart Data Access* - whether you use an existing source system or create a new one.

   More information:

   - Selecting the Source for Type Database Table or View [page 218]
   - Selecting the Source for Type Virtual Table Using SAP HANA Smart Data Access [page 220]

- Selecting the Source for Type DataSource (BW) [page 223]
- Selecting the Source for Type Transformation [page 223]
- Selecting the Source for Type DataStore Object (advanced) [page 224]

5. Once you have completed all wizard steps, the Open ODS View editor opens. In the left of the Semantics tab page, the system now displays the source object of the selected new source object. The assignment of source field to field of the Open ODS views is carried out automatically by determining that the names are identical.
6. Once you have made sure that all fields in the Open ODS view are assigned to suitable fields in the source object, you can save and activate the Open ODS view.

4.3.8 Creating a Data Flow for Open ODS Views

For Open ODS views with source type *Database Table or ViewVirtual Table Using SAP HANA Smart Data Access* or *DataSource (BW)*, you can create a data flow with a DataStore object (advanced) or a transformation. When the data flow is created, the source of the Open ODS view is automatically replaced with the created object.

Context

Data flow with DataStore object (advanced): You can create a DataStore object (advanced) with data flow as data persistency for the Open ODS view. In this case, the scenario is similar to a Persistent Staging Area (PSA) in SAP BW/4HANA. The data can then be processed further in SAP BW/4HANA from within the DataStore object (advanced). For reporting purposes and for using the CompositeProvider, we recommend using the Open ODS view directly.

Transformation: You can use the transformation with data flow to perform data type conversions, assignments or string operations, for example on the source data of an Open ODS view.

For Open ODS view with a database table or a virtual table as a source, the data flow is generated in two steps: the first step is to generate a DataSource, which can be used as a source for the Open ODS view. The second step is to create the target object with the associated data flow.

Creating DataStore Objects (Advanced)

Procedure

1. On the *General* tab, choose *Generate Dataflow* under *Semantics*.
   The wizard for creating the dataflow appears.
2. If the Open ODS view has a database table or a virtual table as a source, create a DataSource in the first step:
   a. Assign a DataSource name.
   b. Specify what to do with the data types of the fields.

The analytic manager only supports specific data types for reporting. If the source uses different data types, these types are automatically converted at runtime.

If you create the target object with SAP BW/4HANA data types, this conversion does not need to be performed at runtime. In addition, this makes it easier to use the SAP BW/4HANA target objects in other BW objects.

c. Choose *Finish*.

- The system now creates a DataSource in the source system of the Open ODS view in application component NODESNOTCONNECTED. The fields of this DataSource are created from the source fields of the Open ODS view .
- The system replaces with Open ODS view with the source object. The source object is now the DataSource.

3. If the Open ODS view now has a DataSource as the source, create a DataStore object (advanced): To do this, go to the *General* tab and choose *Generate Dataflow* under *Semantics*.

   The wizard for creating the dataflow appears. Enter the target object that you want to generate a data flow for here.

   a. Select the object type *DataSource object (advanced)* as the target object.
   b. Assign a name for the DataStore object (advanced).
   c. Specify what to do with the data types of the fields.

      The analytic manager only supports specific data types for reporting. If the source uses different data types, these types are automatically converted at runtime.

      If you create the target object with SAP BW/4HANA data types, this conversion does not need to be performed at runtime. In addition, this makes it easier to use the SAP BW/4HANA target objects in other BW objects.

   d. Choose *Finish*.

Results

- The system creates a DataStore object with an active table in the InfoArea of the open ODS view. The fields of this table are transferred from the fields of the structure of the Open ODS view. Associations to Open ODS view fields are ignored when the DataStore object is created.
- The system replaces with Open ODS view with the source object. The source object is now the DataStore object.
- The system creates a transformation between the DataSource and the DataStore object and a data transfer process for this transformation.

Creating Transformations

Procedure

1. On the *General* tab, choose *Generate Dataflow* under *Semantics*.
   The wizard for creating the dataflow appears.

2. If the Open ODS view has a database table or a virtual table as a source, create a DataSource in the first step:
    a. Assign a DataSource name.
    b. Specify what to do with the data types of the fields.

        The analytic manager only supports specific data types for reporting. If the source uses different data types, these types are automatically converted at runtime.

        If you create the target object with SAP BW/4HANA data types, this conversion does not need to be performed at runtime. In addition, this makes it easier to use the SAP BW/4HANA target objects in other BW objects.
    c. Choose *Finish*.

    - The system now creates a DataSource in the source system of the Open ODS view in application component NODESNOTCONNECTED. The fields of this DataSource are created from the source fields of the Open ODS view .
    - The system replaces with Open ODS view with the source object. The source object is now the DataSource.
3. If the Open ODS view now has a DataSource as the source, create an InfoSource as the transformation target: To do this, go to the *General* tab and choose *Generate Dataflow* under *Semantics*.

    The wizard for creating the dataflow appears. Enter the target object that you want to generate a data flow for here.

    a. Select the object type *InfoSource* as the target object.
    b. Assign a value for the InfoSource.
    c. Specify what to do with the data types of the fields.

        The analytic manager only supports specific data types for reporting. If the source uses different data types, these types are automatically converted at runtime.

        If you create the target object with SAP BW/4HANA data types, this conversion does not need to be performed at runtime. In addition, this makes it easier to use the SAP BW/4HANA target objects in other BW objects.

        If you want to create the target object close to the source, with the data type of the source that is, select the *Source Object Types* option. If you want to create a target object that is close to BW (in other words, with SAP BW/4HANA BW data types), select the *BW Data Types* option.
    d. Choose *Finish*.

Results

- The system creates an InfoSource in application component NODESNOTCONNECTED. The fields of this InfoSource are copied from the fields of the structure of the Open ODS view. Associations to Open ODS view fields are ignored when creating the InfoSource. The InfoSource is based on fields.
- The system creates a transformation between DataSource and InfoSource.
- The system replaces with Open ODS view with the source object. The source object is now the transformation.

Related Information

Creating DataStore Objects [page 273]
Transformation of Data from External Sources for Open ODS Views [page 54]

4.4 Creating InfoObjects

You can create an InfoObject that can be used either in InfoProviders or as an InfoProvider in its own right.

Context

InfoObjects are displayed in the BW Modeling Tools in the BW Repository tree or under the specified InfoArea in the *Characteristic*, *Key Figure*, *Time Characteristic*, *Unit* and *XXL InfoObject* folders.

Procedure

1. You are in the BW Modeling tools. In the context menu for your BW project, choose *New* > *InfoObject*.

   You can also create InfoObjects from the data flow. To do this, go the *BW Object Types* in the editing screen for your data flow, select the type *InfoObject* and drag it to *Details*. Then open the context menu for this InfoObject and choose *Create Persistent BW Object*.

   You will now be guided through the creation process by a wizard.
2. Select an InfoArea. To ensure that you can find your InfoObject again without difficulty, we recommend setting the *Add to Favorites* flag.
3. Enter a technical name and a description. The technical name can be between three and nine characters in length.
4. By choosing *Copy from...*, you can select an object template. This allows you to copy over the properties of this object to your new InfoObject and to edit them as required.
5. You can reference an InfoObject. This gives the new InfoObject the technical properties of the reference object. You can only edit these properties in the reference InfoObject however.
6. You can choose from the following InfoObject types: Characteristic, key figure, unit and XXL InfoObject. Depending on the type you choose, the data type selection will change as well. The InfoObject type cannot be changed subsequently. The data types DATS and TIMS cannot be changed subsequently either.

   If you use referenced characteristics, the data type is copied over. If you use referenced key figures, the key figure type is taken over.
7. When you have finished making your settings, choose *Finish*. The editor appears.
8. Now proceed according to the description for your InfoObject type.

Related Information

Creating Characteristics [page 238]
Creating Key Figures [page 265]
Creating a Unit [page 269]
Creating XXL InfoObjects [page 269]
Reference InfoObjects [page 64]
InfoObject [page 54]

4.4.1 Creating Characteristics

You can create a characteristic that serves as a sort criterion, for example, company code, product, customer group, fiscal year, period or region.

Context

Procedure

1. You are in the BW Modeling tools. You have created a new characteristic using the wizard and you are in the editor.
2. Edit the general settings on the *General* tab page.
3. If you have set the property *Master Data* and/or *Texts*, then you can now edit the settings for master data or texts on the *Master Data/Texts* tab.
4. If you have set the *Hierarchies* property, you can now edit the settings for the hierarchies on the *Hierarchies* tab.
5. On the *BI Clients* tab you can configure the settings for the BI clients.
6. You can edit the attributes and XXL attributes on the *Attributes* tab.
7. On the *Extended* tab page, you can make various additional settings:
8. If you have set the property *Usable as InfoProvider* on the *General* tab, you will see an additional tab: *Runtime Properties*. Here you can perform various settings at query runtime. More information: Query Properties
9. Choose *Miscellaneous* to open either the editing screen for master data or transfer routines (if the characteristic has master data and/or texts) or to open the editing screen for hierarchies (if the characteristic has hierarchies).

> **Note**
>
> When you create a transfer routine, it is valid globally for the characteristic and is included in all the transformation rules that contain the InfoObject. The transfer routine is only run in one transformation

however, with a DataSource as the source. The transfer routine is only used to correct data before it is updated in the characteristic. During data transfer, the logic stored in the individual transformation rule is executed first. The transfer routine for the value of the corresponding field is then executed for each InfoObject that has a transfer routine. In this way, the transfer routine can store InfoObject-dependent coding that only needs to be maintained once and is then valid for all transformation rules.

10. Under *Properties* in the screen area under the Editor, you can see information about the DDIC elements, for example, and the number range objects of the characteristic. You can navigate further to the relevant information via the links.

> **Note**
>
> The generated view are only intended for internal use, not for use as an interface. You can use the external SAP HANA view instead, or official interfaces. More information:

Related Information

Editing General Settings [page 239]
Editing the Properties of Master Data/Texts [page 246]
Editing Properties of Hierarchies [page 248]
Editing the Settings for BI Clients [page 251]
Editing Attributes [page 249]
Editing Various Settings [page 264]

4.4.1.1 Editing General Settings

You can define the characteristic's general properties here.

Context

You are on the *General* tab. For referenced characteristics, many properties are displayed read-only as they cannot be changed.

Procedure

General

1. You can specify whether an attribute view and/or an analytic view should be created for your characteristic in the SAP HANA database. More information:

Dictionary

2. You can edit the data type and the data length.
3. You can specify a conversion routine. More information: Conversion Routines in the BW System

   The conversion routine SNUMC is available for data in integer format (INT4, INT8, DEC). More information: Characteristics of Type Integer [page 63]
4. You can set the *High Cardinality* flag. The High Cardinality property allows you to create more than two thousand million characteristic attributes for a characteristic. As a characteristic with high cardinality has a number of restrictions, you should only set this property if you really expect a larger number of characteristic attributes. More information: High Cardinality [page 56]
5. If you set the *Case Sensitive* flag, the system differentiates between uppercase letters and lowercase letters when values are entered. If it is not set, the system converts all letters into uppercase letters. No conversion occurs during the load process or during transformation. This means that values with lowercase letters cannot be posted to an InfoObject that does not allow lowercase letters.

   > **Note**
   >
   > If you allow lowercase letters, be aware of the system response when you enter variables:
   >
   > If you want to use the characteristic in variables, the system is only able to find the values for the characteristic if the lowercase letters and the uppercase letters are typed in accurately on the input screen for variables. If you do not allow the use of lowercase letters, any characters that you type in the variable screen are converted automatically into uppercase letters.

Properties

6. You can specify whether the characteristic should have master data and/or texts. You can then see the *Master Data/Texts* tab page and can edit the properties displayed there.
7. If the characteristic has master data and/or texts, you can choose *Enhanced Master Data Update*. More information: Master Data Update [page 58]

   If data has already been loaded into the characteristic, you can not activate *Enhanced Master Data Update*.
8. You can specify whether the characteristic should have hierarchies. You can then see the *Hierarchies* tab page and can edit the properties displayed there.
9. If you set the flag *Usable as InfoProvider*, then queries can be executed on the characteristic and the flag can be used in a CompositeProvider. Then the additional tab *Runtime Properties* appears, where you can configure various settings at query runtime. More information: Query Properties

   For InfoObjects as InfoProviders, you can execute a standard query by choosing *BW Reporting Preview* in the lower screen area.
10. If you flag the characteristic as authorization-relevant, the authorization check will run whenever the query is worked on. Set the Authorization-Relevant flag for a characteristic if you want to create authorizations that restrict the selection conditions for this characteristic to single characteristic values.
11. If you select *Attribute Only*, the characteristic can be used only as a display attribute for another characteristic, not as a navigation attribute.
12. You can specify whether an ODATAV4 service is to be provided for the InfoObject. For more information, see Generate ODATAV4 Service for an InfoObject [page 241].

Compounds

13. Under *Compounds*, you can compound the characteristic with other InfoObjects. Compounding of InfoObjects can be necessary in order to uniquely identify the characteristic. Choose *Edit* to create a compound.

> **Note**
>
> If storage location A for plant B is not the same as storage location A for plant C, you can only evaluate the characteristic *Storage Location* in connection with once plant. In this case, compound characteristic *Storage Location* with *Plant* to make the characteristic unique.

4.4.1.1.1 Generate ODATAV4 Service for an InfoObject

Generating an ODATA service for an InfoObject allows you to access information about this object using a URI without having to log on to the SAP BW/4HANA system. The data is displayed in the ODATAV4 Web protocol standard.

Prerequisites

ODATAV4 services can be generated for InfoObjects that meet the following conditions:

- The InfoObject is a characteristic.
- The InfoObject is **not** a reference object.
- The InfoObject does **not** only consist of texts. In other words the InfoObject must have at least one attribute.
- The InfoObject does **not** have any geo-attributes.
- The InfoObject does **not** belong to one of the technical InfoObjects delivered by SAP (RSDIOBJFIX).

If the InfoObject has hierarchies, these **cannot** be read by the ODATAV4 service.

The ODATAV4 service is based on a CDS view that combines the modeled properties for compounded characteristics, master data, and texts of the InfoObject in a flat structure. If the master data contains further InfoObjects, the ODATAV4 service only contains the definition of these InfoObjects, not their attributes and texts. To obtain this data, the ODATAV4 service can be used for these InfoObjects.

Authorizations

Users need certain authorizations to read data of the InfoObject using the ODATAV4 service.

Default Authorizations at InfoObject Level

The default setting in Customizing is for **authorizations to be assigned using access controls**. The system uses authorization object `S_RS_IOBJA` to control access to the data. This allows you to control the authorization for each InfoObject and InfoArea. This means: Users with access authorization can see all the data of the InfoObject. Users without access authorization cannot see any data.

An authorization for attribute values of the InfoObject **cannot** be made with this approach.

When the ODATA service is called, the system also checks authorization object `S_START`. The parameters for this authorization are `R3TR G4BA` (ServiceBinding) and the name of the service binding for the InfoObject.

Analytical Authorizations at Attribute and InfoObject Level

Analytical authorizations allow you to assign authorizations using authorization object `S_TABU_LIN` and transation `RSECADMIN` based on the values of the attributes of the InfoObject. This means: Users without

access authorization cannot see any data. If a user has access authorization, you can use these additional authorizations to define which data of the InfoObject the user can see. You can specify for example that the user is only allowed to see the data from one profit center.

Transitioning to Analytical Authorizations

You can also transition to analytical authorizations for existing InfoObjects mit ODATA services that have previously had default authorizations assigned to them using access control. A special transition procedure is required for this. This is explained in detail here: Changeover to Analytical Authorizations for Generated ODATAV4 Services [page 244].

In Customizing table `RSCXO_ODATA_AUTH`, you can define globally which procedure is used to assign the authorizations. This setting applies to all InfoObjects with ODATA service. When an InfoObject is regenerated, the system immediately applies the Customizing setting.

Context

Time-Dependent Texts:

If the InfoObject has time-dependent parts (texts and attributes), you have to specify a time interval using a parameter in the CDS view. (If you want to obtain all data, you have to enter 01.01.1001 as the lower interval limit and 31.12.9999 as the upper interval limit.)

Example

- Example for reading all time-dependent data with filter:
  ```
  /sap/opu/odata4/sap/<Service_Gruppenname>/srvd_a2x/<ServiceBinding>/0001/
  <ServiceDefinition>(p_business_date_from=1001-01-01,p_business_date_to=9999-1
  2-31)/Set?$filter=(<Attribut>eq '*003' and <Attribut> eq '*3000')
  ```
  The service group, service binding, and service definition are derived from the InfoObject name.
- Example for reading all time-dependent data of a year with filter:
  ```
  /sap/opu/odata4/sap/<Service_Gruppenname>/srvd_a2x/<ServiceBinding>/0001/
  <ServiceDefinition>
  (p_business_date_from=2021-12-17,p_business_date_to=2022-12-17)/ Set?
  $filter=(<Attribut>eq '*003' and <Attribut> eq '*3000')
  ```

Language Dependency

If the InfoObject has language-dependent parts, selections cannot be made using a parameter. An error message therefore appears if a data record with multilingual texts is read with keys. Reading without keys works without restrictions however. You can either filter the required languages or omit the filter for displaying all languages.

Example

- Example of a filter for language English:
  ```
  /sap/opu/odata4/sap/<Service_Gruppenname>/srvd_a2x/<ServiceBinding>/0001/
  <ServiceDefinition>
  (p_business_date_from=2021-12-17,p_business_date_to=2022-12-17)/ Set?
  $filter=(<Attribut>eq '*003' and <Attribut> eq '*3000' and langu_txt = ‘EN’)
  ```
- Example for all languages:

/sap/opu/odata4/sap/<Service_Gruppenname>/srvd_a2x/<ServiceBinding>/0001/<ServiceDefinition> (p_business_date_from=2021-12-17,p_business_date_to=2022-12-17)/Set?$filter=(<Attribut>eq '*003' and <Attribut> eq '*3000')
The system returns all languages.

Delete

If you delete an InfoObject, the system deletes all ODATAV4 artifacts generated for it.

Procedure

Generating an ODATAV4 Service by Activating the InfoObject

1. Call the InfoObject modeling transaction in the BW modeling tools.
2. On the *General* tab page, set the *Generate ODATA Service* flag.
3. When you have finished modeling, activate the InfoObject.

   The system generates an ODATAV4 service based on this modeling. Under *Properties*, you can find all artifacts that are required to generate the ODATAV4 service. These are as follows:

   - CDS view for modeling the InfoObject
   - CDS view for modeling the texts
   - Access control for handling the authorization to read the data
   - Service definition to expose the entities of the CDS view
   - Service binding that contains the URL of the ODATAV4 service.

   The ODATAV4 service is published automatically when the InfoObject is activated. The service is published using a service group that has the same name as the service binding. The associated repository object is SRVD_A2X.
4. You can also make changes to an InfoObject that ODATAV4 services have already been generated for. When the InfoObject is reactivated, the ODATAV4 service is generated in accordance with the current definition. Note however that this also occurs if the changes cause an incompatible change for the ODATAV4 service.

   > ⓘ Note
   >
   > Note how the system reacts if you deselect the flag in the BW modeling tools: Alll generated artifacts are deleted apart from ones created using a self-service. Deletion still take place even if the current version of the InfoObject contains an incompatible change or the new data of the InfoObject forbids an ODATAV4 service. If incompatible changes exist, the ODATAV4 artifacts are regenerated only for ODATAV4 self-services.
5. Transport the InfoObject.

   The transport of the ODATAV4 artifacts takes place exclusively using the TLOGO object InfoObject. The artifacts are generated in the target system using an after-import method.

   > ⓘ Note
   >
   > You can also generate the ODATAV4 service directly using a self-service without having to transport the InfoObject.

Related Information

Changeover to Analytical Authorizations for Generated ODATAV4 Services [page 244]

4.4.1.1.2 Changeover to Analytical Authorizations for Generated ODATAV4 Services

In the case of existing InfoObjects with ODATA services for which standard authorizations have been assigned in the past, you can now change over to controlling access using analytical authorizations instead. The changeover procedure is described below. Following the changeover, the required filters and the ODATA artifacts are generated automatically for both existing and new InfoObjects. It is then no longer necessary to execute the described reports.

Context

Note that it is necessary to perform the individual tasks in exactly the same sequence as described here.

Procedure

InfoObject Settings

1. On the *Master Data/Texts* tab under *Configure Authorization*, choose *Master Data Maintenance with Authorization Check*. If this flag is set, the system also uses authorization object S_TABU_LIN to check which characteristic values the user has authorization for.

   > ⓘ Note
   >
   > To make sure that the S_TABU_LIN entries are processed correctly during replication, see SAP Note 3319042.

2. On the *General* tab page under *Properties*, check that the InfoObject is used as an InfoProvider and is flagged as authorization-relevant.
3. Maintain the values for the authorizations in *Analysis Authorization Management* (transaction `RSECADMIN`).
4. In addition to the authorizations covered by S_TABU_LIN, you can specify in Customizing that the analysis authorizations should also be taken into account.
5. Transport the InfoObject settings.

   Existing ODATA services with default authorizations will still work.

Settings for Users and Authorizations

6. Maintain all users with the appropriate authorizations in the relevant systems.

Change Customizing: Analytical Authorizations for InfoObjects

7. For every InfoObject that you want analytical authorizations to be applied for, maintain the Customizing activity *SAP BW/4HANA* > *Data Procurement* > *Change Authorizations for Master Data Maintenance*.

Create the Filter Based on Users and Authorizations

8. Run program `RS_ODATA_RSEC` to generate the filters for all InfoObjects and all users.

   The system selects all InfoObjects that have an ODATA flag. If analytical authorizations are required, the InfoObject must also be used as an InfoProvider.

   The system selects all users who have authorization `S_RS_IOBJA` for the InfoArea of one of the selected InfoObjects as well as authorisation `S_START` for a service dinging of an ODATA service.

   The system replicates the authorizations set for these InfoObjects and users (`S_TABU_LIN` and/or `RSERCADMIN`) to a filter table. During execution of the ODATA service, these filteres ensure that users only see data that they have authorization for.

   Schedule this report to run when the authorizations are replicated for the first time and also whenever the authorizations are changed. You can insert the report into a process chain. To do this, use process type `RS2ODATA`.

   You can use the *Garbage Authorizations* processing mode to delete authorizations that are not required any longer, for example in cases where the InfoObject or user no longer exists in the system. If required, you can execute this processing mode at greater intervals. The authorizations awaiting deletion do not impair the functionality of the application.

Customizing Change for the ODATA Authorization Procedure

9. Set the analytical authorizations in Customizing.

   Existing ODATA services continue to run as before. The changed behavior only takes effect during regeneration.

Generate ODATA Artifacts

10. Run program `RSXCO_GENERATE_ODATA` to generate the ODATA artifacts.

    The system checks whether there is an entry for the corresponding users in the filter table for each InfoObject to be generated. If this is not the case, an error is issued, and the report terminates processing.

    You can still run the report by selecting the *Ignore Missing Filter Entries* parameter. Note however in this case that authorizations might not be correctly applied

    > ⚠ Caution
    >
    > Note the sequence: First replicate the filters (see step 8) before you run the program for generation.

Results

If existing InfoObjects are generated or if new ones are created after changing over to analytical authorizations, the system automatically generates the required filters and the ODATA artifacts. In this case it is no longer necessary to run the programs described above after generating the artifacts.

Next Steps

In the case of authorization changes, we recommend running program periodically RS_ODATA_RSEC for either all InfoObjects and all users or for specific users and InfoObjects.

Related Information

Editing General Settings [page 239]
Editing the Properties of Master Data/Texts [page 246]

4.4.1.2 Editing the Properties of Master Data/Texts

Here you can edit the properties for master data/texts.

Context

The *Master Data/Texts* tab is displayed if you have set the property *Master Data* or *Texts* on the *General* tab.

Procedure

Read Access

1. If required, you can change the master data access. We recommend using the default setting.

   You have the following options for accessing master data at query runtime:

   - Standard: The values from the characteristic's master data table are displayed. This is the default setting.
   - Own Implementation: You can define an ABAP class to implement the access to master data yourself. To do this, you need to implement interface IF_RSMD_RS_ACCESS. You also have to be proficient in ABAP OO. An example of this is time characteristic 0FISCYEAR, which is delivered with the the Business Content.
   - Direct: This enables you to access the data in a source system directly. If you choose this option, you have to use a data transfer process to connect the characteristic to the required DataSource. You also have to assign the characteristic to a source system.
   - SAP HANA View: You can use this option to create virtual master data.

     > ⓘ Note
     >
     > Note: To achieve optimum reporting performance when using master data (text, attributes, hierarchies), we recommend using InfoObjects with persistent attribute, text and hierarchy tables instead of virtual access to master data via SAP HANA views. As it is not possible to guarantee

referential integrity on SAP HANA views, a considerable number of SAP HANA optimizations (push down) in the SAP BW/4HANA Analytic Engine are not possible with virtual access to master data.

If you intend to model a virtual access to master data, we still recommend using Open ODS views as the central object for accessing data from SAP BW/4HANA. Note however that restrictions to Open ODS views might apply, which could force you to continue using InfoObjects based on SAP HANA views:

- Hierarchies are not supported in Open ODS views.
- Key date-based access to time-dependent master data is not supported in the Open ODS views
- In the case of planning scenarios with CompositeProvidern based on field-based DataStore objects (advanced), navigation attributes cannot from Open ODS views cannot be used.
- The implementation of authorizations in InfoObjects and associations to Open ODS views in CompositeProviders is different.

Texts

2. If the characteristic contains texts, you have to select at least one text. The *Short Text* (20 characters) option is set by default, but you can also choose *medium-length* texts (40 characters) or *long texts* (60 characters). If you choose the *Long text is extra long* option, the long text is of type SSTRING and can have up to 1333 characters.
3. You can specify whether or not the texts in the text table are language-dependent. If you specify language-dependent, the language is a key field in the text table. Otherwise, there is no language field in the text table.
4. You can specify whether the texts are time-dependent. If so, the date is included in the key of the text table.

Data Target

5. You can assign an application component, in order to

   - use a characteristic as an InfoSource using direct update. The system displays the characteristic in the InfoSource tree in the Data Warehousing Workbench. You can assign DataSources and source systems to the characteristic here. You can then also load attributes, texts, and hierarchies for the characteristic. More information: 3.x InfoSource
   - Generate an export DataSource for a characteristic. To be able to do this, the characteristic must also be an InfoSource with direct updating.

6. If you set the *Permitted for Real-Time Data Acquisition* flag, you can use real-time data acquisition to fill the characteristic with data.

Configure Authorization

7. If you set the *Master Data Configuration with Authorization Check* flag, you can use authorizations to protect the attributes and texts for this characteristic from being maintained at single-record level. If you activate this option, you can enter the characteristic values for which the user has authorization for each key field in the master data table. You do this in the profile generator in role maintenance using authorization object S_TABU_LIN. If you do not set this flag, you can only assign the authorization for master data maintenance as a whole (for all characteristic values).

4.4.1.3 Editing Properties of Hierarchies

Here you can edit the properties for hierarchies.

Context

If you have set the *Hierarchies* property on the *General* tab, the system will show the *Hierarchies* tab.

The system then generates a hierarchy table with hierarchical relationships for the characteristic.

Procedure

Hierarchy Type

1. If you want to use various versions of the characteristic hierarchy in the query, you can set the *Version-Dependent* flag.
2. If you want the entire hierarchy or the hierarchy structure to be time-dependent, you can select the *Time-Dependent* flag.

Time-Dependent Properties

3. If you have set the *Time-Dependent* flag, you can specify further entries here.
    - If you want to create views of a hierarchy that are valid for a specific period of time, you can specify the entire hierarchy as time-dependent.
    - If you want to model relationships that change time-dependently, you can specify the hierarchy structure as time-dependent.

Miscellaneous

4. You can define whether intervals in hierarchies are permitted for the hierarchy basic characteristic. Further information: Intervals
5. You can define whether the sign reversal function is possible for the hierarchy basic characteristic. If reversing the sign is activated for hierarchies, the attribute 0SIGNCH is included in the communication structure for hierarchy nodes.

    In hierarchy maintenance, you can specify whether the sign for transaction data booked to this node is to be reversed for each hierarchy node. In the query definition, create a formula variable for reversing the sign.

    > ### Example
    >
    > You have a hierarchy based on receipts and expenditure. According to accountancy logic, income is displayed with a negative sign and expenses are shown with positive sign. Adding these values produces the profit.
    >
    > Under certain circumstances, it may be advisable to circumvent accountancy logic and display the query income with a positive sign. This is where reversing the sign comes in. In our example, one would only activate sign reversal for income.

External Characteristics in Hierarchies

6. You can specify which characteristics are permitted in the hierarchy nodes. If you do not select an InfoObject here, only text nodes are permitted as nodes that can be posted to in hierarchies.

4.4.1.4 Editing Attributes

Here, you can create display, navigation and XXL attributes for the characteristic and edit its properties.

Context

This tab page is only available if you have set the *Master Data* flag on the *General* tab page.

Procedure

Navigation and Display Attributes

1. Under *Display and Navigation Attributes*, you can create a new attribute by choosing *Add*. Select an InfoObject and choose *OK*. The attribute is applied. Via the link on the attribute name, you can call the InfoObject maintenance screen. On the right side of the screen, you can edit the properties of the attribute.

   Note the conditions for time characteristics: Navigation Attributes for Time Characteristics [page 250]

General

2. You can choose whether the attribute is a display attribute (DIS) or a navigation attribute (NAV). The attributes are defined as display attributes by default.

Properties

3. You can determine the sequence in which the attributes for a characteristic are displayed in *F4 input help*.

   There are the following values for this setting:

   - 00: The attribute is not displayed in input help
   - 01: The attribute appears in first position (far left) in input help
   - 02: The attribute appears in second position in input help.
   - 03: ...

   The total number of fields allowed in input help is restricted to 40. In addition to the attributes, the characteristic itself, its texts, and the compound characteristics are generated in input help. The total number of fields cannot be greater than 40.

4. You can specify whether the attribute is time-dependent. Even if only one attribute is time-dependent, a time-dependent master data table is created. There can still be time-independent attributes for this characteristic however. These are stored in a separate table.
5. By choosing *Part of Index*, you stipulate whether the attribute should be added to the secondary index that is defined on the master data tables (P/Q/X/Y) of a characteristic.

Navigation Attribute Properties

6. You can enter a description for the navigation attribute. This can be useful if a characteristic appears more than once in an InfoProvider (as a characteristic and as a navigation attribute). You can thus differentiate between these in the query definition.
7. You can mark navigation attributes as authorization relevant independently of the assigned basic characteristics.
8. By choosing *Maintain transitive attributes* in the context menu for an attribute, you can define the transitive attributes for the attribute in question. This allows you to make the attributes of the query available for navigation in the query.
9. By choosing *Remove*, you can remove attributes from the characteristic again.

XXL Attributes

10. Under *XXL Attributes*, you can create a new XXL attribute by choosing *Add*. Select an InfoObject and choose *OK*. The XXL attribute is applied.

4.4.1.4.1 Navigation Attributes for Time Characteristics

Navigation attributes can be specified for time characteristics.

The following is a list of permitted navigation attributes:

Time Characteristic	Permitted Navigation Attributes
0DATE	0CALWEEK
	0CALMONTH2
	0HALFYEAR1
	0CALQUARTER
	0CALQUART1
	0WEEKDAY1
	0FISCVARNT*
	0FISCPER*
	0FISCYEAR*
0CALMONTH	0CALMONTH2
	0CALYEAR
	0CALQUARTER
	0CALQUART1
	0HALFYEAR1

Time Characteristic	Permitted Navigation Attributes
0CALQUARTER	0CALQUARTER
	0CALYEAR
	0HALFYEAR1
0FISCPER	0FISCYEAR
	0FISCPER3

* Unlike other characteristics, it is possible (or in fact necessary) in this case to add referencing characteristics. Only fiscal variants/periods/years with a constant fiscal variant are used.

The navigation attributes also have to be activated on the referencing characteristic 0CALDAY and can also be activated for other referencing characteristics that are based on 0DATE,0CALMONTH or 0FISCPER.

The fiscal year time characteristics (FISC*) must be created on the basis of a constant fiscal year variant. For every fiscal year variant, a fiscal year group must be created with a reference to 0FISCVARNT, 0FISCPER and 0FISCYEAR.

These fiscal year groups are generated because this is then consistent with the master data table, as there is only one value per date.

If a new value is loaded for a time characteristic in an InfoProvider, the master data entries are automatically filled with the derived attribute values.

4.4.1.5 Editing the Settings for BI Clients

You can edit the display properties in BI clients here.

Procedure

General

1. Under *Display* you can specify whether the text in executed query should be displayed, and if yes, which text. This setting can be overwritten in queries.
2. Under *Selection* you can define if and how the characteristic values have to be restricted in queries.
3. If you want the initial value in the query to be displayed sorted (as opposed to at the final position), you can select *Include Initial Value in Sort Sequence*.

Geographical

4. For each geo-relevant characteristic you have to specify a geographical type.

   There are five options to choose from:

   - 0: *No Geographical Data*: This characteristic cannot be used as a geo-characteristic. Geographical attributes of the InfoObject (such as 0LONGITUDE, 0ALTITUDE) are deleted.

- 1: *Shapes*: For this type you can use shapefiles (country borders, for example), to display the characteristic on a map.

  > **Note**
  >
  > Shapefiles are currently not supported by any front end.

- 2: *Point Data (Attributes 0LONGITUDE and 0LATITUDE)*: For this type, geo-attributes are generated that make it possible, for example, to display customers as a point on a map. This option is supported by SAP Analytics Cloud.
- 3: *Derived Point Data from Attribute*: For this type, the geo-attributes of a geo-characteristic of type 2 or type 4, which is the attribute, are used. Here the point data is referenced, not the shapefiles. This option is supported by SAP Analytics Cloud.
- 4: *Shapes and Point Data* This type is a mix of type 1 and type 2. This option is partly supported by SAP Analytics Cloud.

More Information: Shapes and Point Data [page 252]

5. If you have selected the geographical type *Derived Point Data from Attribute* for the characteristic, here you specify the characteristic attribute whose geo-attributes you want to use.

Query Filter Value

6. You can specify how the selection of filter values or the restriction of characteristics is determined when you define a query. When you restrict characteristics, the values from the master data table are usually displayed. For characteristics that do not have master data tables, the values from the SID Table are displayed instead. In many cases it is more useful to only display those values that are also contained in an InfoProvider. Therefore you can also choose the setting *Only Values in InfoProvider*.
7. You specify how the selection of filter values is determined when a query is executed. When queries are executed, the selection of filter values is usually determined by the data that is selected by the query. This means that only the values for which data has been selected in the current navigation status are displayed. However, in many cases it can be useful to include additional values. Therefore you can also choose the settings *Only Values in InfoProvider* and *Values in master data table*. If you make this selection, however, you may get the message "No data found" when you select your filter values.
8. You can specify how the display of filter values is determined when a query is executed. If, for example, the characteristic has very few characteristic values, it may be useful to display the values as a dropdown list box.

4.4.1.5.1 Shapes and Point Data

Shapes and point data describe data with a geographical reference (for example, characteristics such as customer, sales region, country).

Maps are used to display and evaluate this geo-relevant data.

There are four different types of geo-characteristic:

1. *Shapes*
   For a characteristic with this geographical type, geographical data can be uploaded: This data from areas or polygons is stored in shapefiles that define the geometry and the attributes of this geo-characteristic. Country and region are two examples of this kind of geo-characteristic.
2. *Point Data*

For a characteristic with this geographical type, geographical coordinates can be entered. Customer and plant are two examples of these geo-characteristics since they are rooted to one geographical “point” that can be described by an address.
A range of standard attributes are added to this geo-characteristic. These standard attributes store the geographical coordinates of the corresponding object for every row in the master data table. The geo-attributes concerned are:

Technical Name	Description	Data Type	Length
LONGITUDE	Longitude of the location	DEC	15
LATITUDE	Latitude of the location	DEC	15
ALTITUDE	Altitude of the location (height above sea level)	DEC	17
PRECISID	Identifies how precise the data is	NUMC	4
SRCID	ID for the data source	CHAR	4
GEOFEATID	Feature ID of a geographic region	NUMC	15

> **Note**
>
> At present, only the LONGITUDE and LATITUDE attributes are used. ALTITUDE, PRECISID and SRCID are reserved for future use.
>
> If you reset the geographical type of a characteristic to *Not a Geo-Characteristic*, these attributes are deleted in the InfoObject maintenance.

3. *Derived Point Data from Attribute*
   To save you having to geocode every single dynamic geo-characteristic individually, a dynamic geo-characteristic can take its geo-attributes (longitude, latitude, altitude) from another previously geocoded characteristic of type Point Data - type 2 and type 4 - such as postal code. Customer and plant are examples of this type.
   The system treats this geo-characteristic as a regular geo-characteristic that describes a location (geographical information as a point on map). The geo-attributes described above are not added to the master data table on the database level. Instead, the geo-coordinates are stored in the master data table of a regular attribute of the characteristic.

   > **Note**
   >
   > You want to define a geo-characteristic for *Plant* with the postal code as an attribute. The geo-coordinates are generated from the postal code master data table during the runtime.

   > **Note**
   >
   > This method prevents redundant entries from appearing in the master data table.

4. *Shapes and Point Data*

This type is a combination of the types shapes (type1) and point data (type 2). This means that the characteristic is described by a shape and by geographical coordinates. Postal code is an example of this type.

4.4.1.5.1.1 Delivered Geo-Characteristics

With BW Content, SAP BW/4HANA delivers a range of geo-characteristics.

Structure

The following are the most important delivered geo-characteristics:

Static geo-characteristics

Technical Name	Description
0COUNTRY	Country key
0DATE_ZONE	Time zone
0REGION	Region (federal state, province)

Dynamic geo-characteristics

Technical Name	Description
0AP_OF_FUND	Location number
0TV_P_LOCID	IATA location

Dynamic geo-characteristics with values from attributes

Technical Name	Attributes	Description
0BPARTNER	0POSTCD_GIS	Business partner
0CONSUMER	0POSTCD_GIS	Consumer
0CUSTOMER	0POSTCD_GIS	Customer number
0PLANT	0POSTCD_GIS	Plant

Technical Name	Attributes	Description
0VENDOR	0POSTCD_GIS	Vendor

Static geo-characteristics with geo-attributes

Technical Name	Description
0CITYP_CODE	City district code for city and street file
0CITY_CODE	City code for city and street file
0POSTALCODE	Postal/ Zip code
0POSTCD_GIS	Postal code (geo-relevant)

4.4.1.5.2 Using Geo-Relevant Characteristics in SAP Analytics Cloud

You can display SAP BW/4HANA data geographically in SAP Analytics Cloud (SAC).

For geographical characteristics, select the matching geographical type. When talking about geographical types, a distinction is made between shapes and point data. More Information: Shapes and Point Data [page 252]

When using geographical characteristics, pay attention to a number of points so that these characteristics can be used in SAC: Special Features in the Modeling of Geo-Characteristics [page 256]

In SAP Analytics Cloud, you can add other graph types and a geomap to your story. You can choose from different display types: bubble layer, HeatMap, FlowLayer, and choropleth layer. More Information:

With the choropleth layer, you can use the regional data from the SAP HANA Delivery Unit, or you can upload your own shapes and define your own hierarchy.

Prerequisites

You have installed BW Modeling Tools 1.23 and SAP Analytics Cloud 2020.06.

You need to configure the SAP BW/4HANA system accordingly: Configuring the Use of Geo-Characteristics in SAC [page 256]

Related Information

Configuring the Use of Geo-Characteristics in SAC [page 256]

Special Features in the Modeling of Geo-Characteristics [page 256]
Geographical Reporting in SAP Analytics Cloud (SAC) [page 258]

4.4.1.5.2.1 Configuring the Use of Geo-Characteristics in SAC

If you want to use geo-characteristics in SAP Analytics Cloud (SAC), note the following prerequisites and manual configuration steps.

Prerequisites

You are running SAP HANA Version 122.06. More information: SAP Note 1600929

Procedure

1. Implement the *Spatial Data Delivery Unit for SAP Analytics Cloud* containing the regional data for the Choropleth layer. More information: SAP Note 2395407 This contains the following steps:
   - Create the *spatial reference system* 3857.
   - Import the.tgz files.
2. Activate the BI Content InfoObjects 0GSHAPE3857 and 0GEOFEATID in precisely this sequence. 0GEOFEATID corresponds to the regions in the Spatial Data Delivery Unit for SAP Analytics Cloud.
3. Create the BW hierarchy SAPBWWORLDHIERARCHY for InfoObject 0GEOFEATID using report RSD_GEO_SYNC_HIER. The BW hierarchy is derived from the Spatial Data Delivery Unit for SAP Analytics Cloud.

4.4.1.5.2.2 Special Features in the Modeling of Geo-Characteristics

When modeling geo-characteristics, you need to bear a few things in mind so that they can be used in SAC.

Context

During activation of characteristics of type *Shapes* (type 1), *Point Data* (type 2) and *Shapes and Point Data* (type 4), a geotable with SAP HANA spatial type ST_POINT is automatically generated. The SAP HANA spatial type is determined from attributes 0LONGITUDE and 0LATITUDE.

For characteristics of type *Shapes* (type 1) and *Shapes and Point Data* (type 4), a field with the SAP HANA Spatial type *ST_GEOMETRY* is also generated and can be used to store shapes.

For characteristics of type *Shapes* (type 1) ST_POINT is not determined by 0LONGITUDE and 0LATITUDE - as for other types - but from the corresponding shape.

When the type is selected, characteristics of type *Point Data* (type 2), *Derived Point Data from Attribute* (type 3) and *Shapes und Point Data* (type 4) are automatically given characteristic 0GEOFEATID as their navigation attribute (type 2 and 4) or transitive attribute (type 3).

> **Note**
>
> Characteristic 0GEOFEATID corresponds to regions of the world as required for the choropleth layer. Attribute 0GEOFEATID is filled with the region that contains the characteristic value geographically and is situated at the lowest position of all regions in the hierarchy. The association in higher regional levels is then created by the hierarchy at runtime.

> **Example**
>
> The region *BEVERLY HILLS* is assigned to a customer geographically. The assignment provides the assignment at higher levels: Beverly Hills - Los Angeles - California - United States.

Characteristics of type Point Data (type 2) and Shapes and Point Data (type 4) are automatically given the attributes LONGITUDE and LATITUDE.

Procedure

1. If you already have geo-characteristics and have upgraded to SAP SAP BW/4HANA SP04, you have to manually add characteristic 0GEOFEATID to these geo-characteristics as the navigation/transitive attribute.
2. When data is loaded into the transformation, navigation attribute 0GEOFEATID must be assigned to the fields 0LONGITUDE and 0LATITUDE. More information: Creating Transformation for Geo-Relevant Characteristics [page 259]
3. The point data is automatically updated in the geotable during master data activation of 0LONGITUDE and 0LATITUDE.
4. For shapes, you have to upload shapefiles. More information: Uploading Shapefiles [page 263]
5. To be able to use the choropleth layer in SAC with BW data, you have to activate navigation attribute 0GEOFEATID in the CompositeProviders that use the queries to be used for geographical reporting. In the queries, you also have to add one or more free dimensions to navigation attribute 0GEOFEATID.

4.4.1.5.2.3 Geographical Reporting in SAP Analytics Cloud (SAC)

This section introduces particular technical aspects of geographical reporting in SAP Analytics Cloud (SAC).

Bubble Layer, HeatMap and FlowLayer

The geomap types Bubble Layer, HeatMap and FlowLayer use the geographical information from the characteristic values, derived from their display attributes 0LONGITUDE and 0LATITUDE.

To ensure satisfactory performance, these attribute values are not evaluated at runtime. These two attributes are converted to the SAP HANA spatial format ST_POINT during activation of the master data instead. This SAP HANA spatial type ST_POINT is used at runtime. Geometric calculations are performed on the basis of this, and location information is converted into th text format required by the front end.

Choropleth Layer

In the choropleth layer, a region is geographically assigned to every characteristic value (0LONGITUDE, 0LATITUDE) at every regional level (federal state for example). The Choropleth layer does not display the geographical locations of the geographical characteristic but the regions of the globe (countries, federal states and so on). The key figure of the region is aggregated using the geographically assigned characteristic values. The regions are therefore a navigation attribute of the geo-characteristic.

Choropleth (Default)

In SAP BW/4HANA, you have to import the regional data via a SAP HANA delivery unit. In SAP BW/4HANA, these regions are represented by the BI Content characteristic 0GEOFEATID. All regions are sorted in the BW hierarchy SAPBWWORLDHIERARCHY (California is thus inserted as a child of USA). Characteristic 0GEOFEATID is added to the geo-characteristic as a navigation attribute.

Attribute 0GEOFEATID is filled with the region that contains the characteristic value geographically and is situated at the lowest position of all regions in the hierarchy. The association in higher regional levels is then created by the hierarchy at runtime.

> Example
>
> The region *BEVERLY HILLS* is assigned to a customer geographically. The assignment provides the assignment at higher levels: Beverly Hills - Los Angeles - California - United States.

At runtime, SAP Analytics Cloud queries the regions as navigation attribute 0GEOFEATID of the geographical characteristic (with active hierarchy SAPBWWORLDHIERARCHY).

Choropleth (Custom)

Instead of use the default hierarchy SAPBWWORLDHIERARCHY, you can create your own hierarchy for your shapes. This enables you to take account of your own sales regions for example. To do this, you define the dependencies of the characteristics when editing your geo-characteristic. These can be multi-level dependencies.

4.4.1.5.2.4 Using Geo-Relevant Characteristics in the Choropleth Layer (Choropleth Default)

You want to use the regional data from the SAP HANA Delivery Unit in order to use geographical characteristics in the choropleth layer.

Procedure

1. You create a characteristic with the geographical type 2, 3, or 4.
2. Create a transformation for 0GEOFEATID or fill 0GEOFEATID for the existing transformation. You do not need this for characteristics with geographical type 3 since filling 0GEOFEATID is done with the geographical attribute characteristic. More information: Creating Transformation for Geo-Relevant Characteristics [page 259]
3. Use the transformation to load the master data.

4.4.1.5.2.4.1 Creating Transformation for Geo-Relevant Characteristics

You are using the regional data from the SAP HANA Delivery Unit.

Context

The regions are assigned by means of a transformation. 0GEOFEATID is filled from 0LONGITUDE and 0LATITUDE.

Procedure

1. Create a transformation for your characteristic.
2. Choose transformation type *Formula* and then choose function *BW_GEO_FEATURE_ID* in formula group *HANA - Geometric Functions*. 0GEOFEATID is filled from 0LONGITUDE and 0LATITUDE.

4.4.1.5.2.5 Using Geo-Relevant Characteristics with Your Own Shape Hierarchy in the Choropleth Layer (Choropleth Custom)

Instead of using the default hierarchy SAPBWWORLDHIERARCHY, you can create your own hierarchy for your shapes.

Context

This enables you to take account of your own sales regions for example. To do this, you define the dependencies of the characteristics when editing your geo-characteristic. These can be multi-level dependencies.

Procedure

1. You create multiple characteristics with geographical type 1 or 4 (with shapes).
2. You create the main characteristic with geographical type 2 or 4, and add the characteristics from step 1 as navigation attributes.
3. Define the hierarchy for the main characteristic. More information: Creating a Hierarchy for Geo-Characteristics [page 260]

4.4.1.5.2.5.1 Creating a Hierarchy for Geo-Characteristics

If a characteristic is defined as geographical, you can define a hierarchy for it.

Context

You can either use the default hierarchy or define one of your own. Every region in a hierarchy is represented by a characteristic with the geographical type *shape* and must be added to the characteristic as a navigation attribute.

Procedure

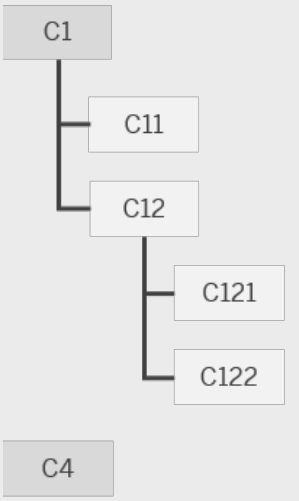
1. 1. Go to the *Attributes* tab page.
2. 2. In the *Depends on* column, specify the characteristic that you want to each characteristic to be dependent on. These can also be multi-level dependencies.

Example

You have defined the following dependencies:

Navi...	Name	Dependent on
☐	[0LONGITUDE] Longitude of the Geo-Loc...	
☐	[0LATITUDE] Latitude of the Geo-Location	
☐	[0ALTITUDE] Geo Location Height	
☐	[0SRCID] Data Source ID of Geo-location	
☐	[0PRECISID] Geo-location precision	
☑	[0GEOFEATID] Feature ID of a geographic...	
☑	[ZHS_C11] C11 for ZHS_SHAHI	ZHS_C1
☑	[ZHS_C121] C121 for ZHS_SHAHI	ZHS_C12
☑	[ZHS_C122] C122 for ZHS_SHAHI	ZHS_C12
☑	[ZHS_C12] C12 for ZHS_SHAHI	ZHS_C1
☑	[ZHS_C1] C1 for ZHS_SHAHI	
☑	[ZHS_C4] C4 for ZHS_SHAHI	

This provides you with the following dependencies, and the following hierarchies are available in SAC: [C1,C11], [C1,C12,C121],[C1,C12,C122],[C4] :



4.4.1.5.2.5.2 Preparing Shapefiles

In order to be able to display shapes geographically, a connection has to be established between the shapefile data and the corresponding characteristic values.

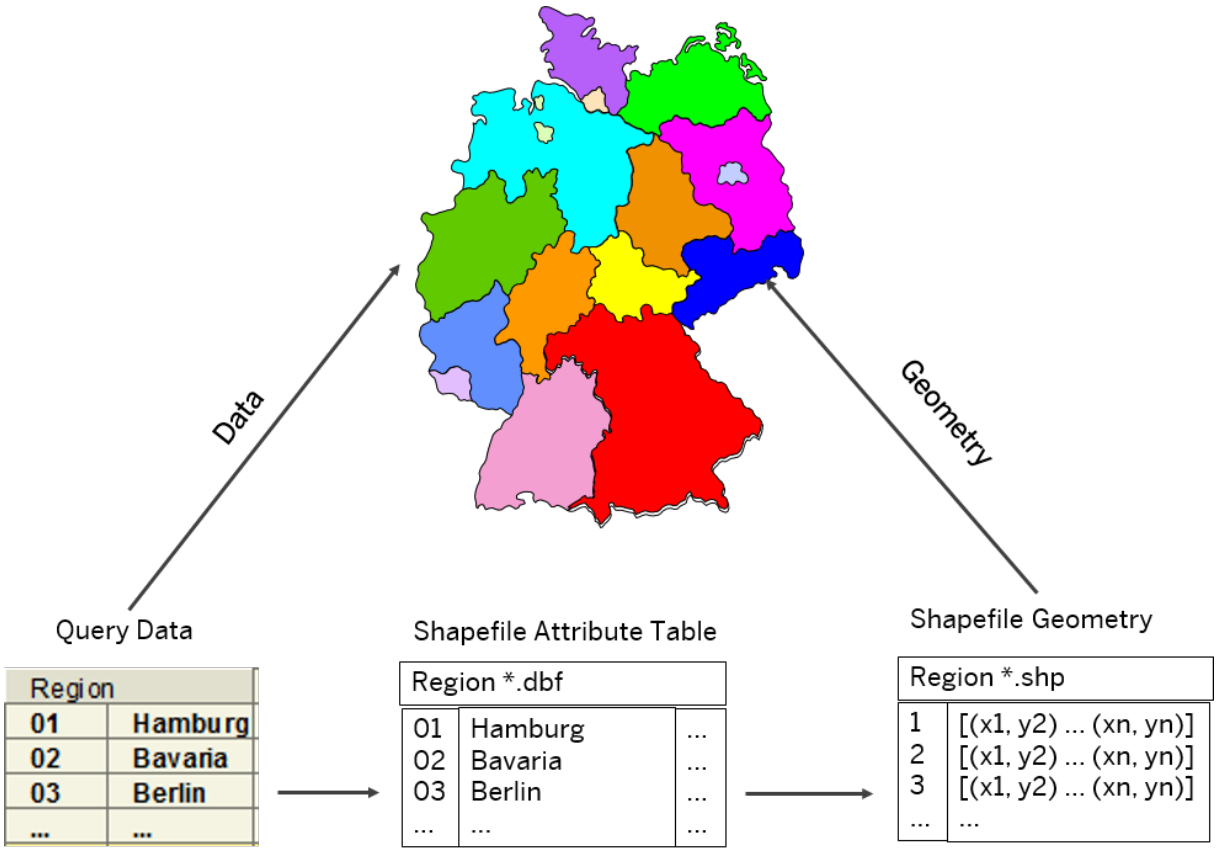
Prerequisites

The characteristic must have the property *With Master Data*. The characteristic values do not have be in the master data yet; they are created automatically.

Context

To assign the entries of the dbase file (dbf) to the characteristic values, you have to add a column to the table.

> ⓘ Note
>
> In previous releases, SAPBWKEY was used for this, but this has changed.



Procedure

1. You are on the *BI Clients* tab in the characteristic editing screen in the BW modeling tools. Choose *Geodata download (everything)* under *Actions*. Choose *Go*. The BW/4 back end opens. You can download the characteristic values in internal format here.
2. Add a column for the characteristic to the table of the dbase file. To assign the entries of the dbase file (dbf) to the characteristic values, you have to add a column with the characteristic name to the table. The type should correspond to the characteristic type, and the characteristic values must be in internal format (example: "0001" instead of "1" for NUMC(4)). For compounded characteristics, you have to add multiple columns.

4.4.1.5.2.5.3 Uploading Shapefiles

Prerequisites

You have added a column or field with the characteristic name to the shapefiles that you want to load shapes for. If using compounding characteristics, you have added multiple columns. More Information: Preparing Shapefiles [page 262]

To update the geotable with shapes that are uploaded to the Business Document Server, you need SAP HANA 2.0 Rev 45.

Procedure

1. You are on the *BI Clients* tab in the characteristic editing screen in the BW modeling tools. You have chosen *Shapes* or *Shapes and Point Data* as the geographical type.
2. Choose *Upload Shapefiles* under *Actions*. Choose *Go*. The BW/4 back end opens.
3. Select your shapefiles. You need the following three shapefiles: .dbf, . shp, .shx. The shapefiles must be in the Spatial Reference ID 4326 and should all have the same name before the ending.
4. Choose *Update Geotable* under *Actions*. Choose *Go*. You can choose either *Delta Update* or *Full Update*.

   The uploaded shapes are transformed and stored in SRID (Spatial Reference ID) 3857. It is therefore not possible to use shapes in SAC if they exceed the limitations of this spatial reference system. This means that they must not have degrees of latitude exceeding 85.06° and -85.06°.

   > ⓘ Note
   >
   > The shapefiles are still uploaded if there are shapefiles there already, but do not overwrite them. If you want to avoid this, you have to delete any existing shapefiles before uploading.

4.4.1.5.2.5.4 Troubleshooting When Updating the Geotable with Shapefiles

When updating the geotable, there are a number of things to look out for. If errors occur, you can check what might have caused this.

Error Message	Action
Invalid table name: Could not find table/view ...	Check whether you have set the *With Master Data* flag for the geo-characteristic. This flag should be set automatically, but could be the cause of the error if you are using an older version of the BW modeling tools or if the characteristic in question has not changed since updating to Release 2.0 SPS04.
Spatial error: Exception 1601801: PROJ.4 transformation error: tolerance condition...	The shapefiles must be uploaded in spatial reference system “WGS 84” with spatial reference ID 4326. The spatial reference system has a number of important limitations: Degrees of latitude exceeding 85.06° and -85.06° are not supported.

Error while handling file <..>.shp	• Every characteristic value should only occur once in all shapefiles.
	• The characteristic values should have the internal format:
	  • In the case of data type NUMC, all entries should be filled with zeros.
	  • Use the characteristic values of the dbf file created by BW/4. More Information: Preparing Shapefiles [page 262]

Reading from .dbf file failed due to missing fields.	To establish the connection between the characteristic values and the entries in the shapefiles, you first need to add a column for the characteristic to the dbf.file. In the case of a compound characteristic, you need to add multiple columns. More Information: Preparing Shapefiles [page 262]

4.4.1.6 Editing Various Settings

You can make various settings for the characteristic here.

Procedure

Unit Conversion

1. You specify a unit InfoObject that is a unit of measure. The unit InfoObject must be an attribute of the characteristic. This unit InfoObject is used when quantities are converted for the master data-bearing characteristic in the Business Explorer. More information: Quantity Conversion
2. You can define units of measure for the characteristic. A DataStore object is created for units of measure. You can specify the name of the quantity DataStore object, the description, and the InfoArea into which the object is to be inserted. The proposal for the name is: UOM<name of InfoObject to which the quantity DataStore Object is added>. More information: Prerequisites for InfoObject-Specific Quantity Conversion
3. You select a unit InfoObject that is a currency that you have created as an attribute for the characteristic. In this way, you can define variable target currencies in the currency translation types. The system determines the target currency using the master data when you perform currency translation in the query and dynamically when loading data.

Miscellaneous

4. You can specify that a characteristic is used as a document property. This enables you to assign a comment or any document to a combination of characteristic values. More information: Documents
5. You can specify a fixed unit of measure.
6. By assigning a constant, a fixed value is assigned to a characteristic. The characteristic then exists on the database (for verifications for example) but does not appear in the query. It is particularly useful to assign constants to compound characteristics.

   > ### Example
   >
   > The *Storage Location* characteristic is compounded with the *Plant* characteristic. If just one plant is always used in the application, a constant can be assigned to this plant. Validation against the storage location master table should then be performed using this value for the plant. In the query, the storage location only appears as a characteristic however.

7. You can specify a DataStore object for checking the characteristic values. If you create a DataStore object for checking the characteristic values in a characteristic, the valid values for the characteristic are determined from the DataStore object rather than from the master data during transformaton. The DataStore object must contain the characteristic itself and all the fields in the compound as key figures.
8. You can make settings for *Read Access Logging (RAL)* . To take into account the characteristic for the *read access logging*, you must enter a *business area* and a *log domain*. You must also activate the characteristic *log read access the property Activate output* for the InfoProvider. More information:

4.4.2 Creating Key Figures

You can create a key figure. Key figures provide the values that are reported on in a query.

Context

For referenced key figures, many properties are displayed read-only as they cannot be changed.

Procedure

1. You are in the BW modeling tools. You have created a new key figure using the wizard and you are in the editor.

Dictionary

2. You can change the key figure type and the data type. For the *amount*, *quantity* and *number*, you can choose either decimal numbers or floating point numbers. For the key figures *Date* and *Time*, you can choose the decimal display to apply to the fields.

> ### Note
>
> Note: Using floating point numbers results in unavoidable rounding errors in all arithmetical operations. We therefore advise against using floating point numbers for amounts and quantities which exact results are required for. SAP Note 2798 describes this problem and illustrates it with the help of a few examples.

The following combinations of key figure and data type are possible:

Key figure type	Data Type
AMO Amount	CURR: Currency field, created as DEC
	FLTP: Floating point number with 8 byte precision
QUA Quantity	QUAN: Quantity field, created as DEC
	FLTP: Floating point number with 8 byte precision
NUM Number	DEC: Calculation field or amount field with comma and sign
	FLTP: Floating point number with 8 byte precision
INT integer	INT4: 4 byte integer, whole number with +/- sign
	INT8: 8 byte integer
DAT Date	DATS: Date field (YYYYMMDD), created as char(8)
	DEC: Calculation field or amount field with comma and sign
TIM Time	TIMS: Time field (hhmmss), created as CHAR(6)
	DEC: Calculation field or amount field with comma and sign

Aggregation

3. You can select an aggregation type.

There are four aggregation options:

- Minimum (MIN) The minimum value of all the values in this column is displayed in the results row.
- Maximum (MAX) The maximum value of all the values in this column is displayed in the results row.
- Total (SUM) The total of all the values in this column is displayed in the results row.
- No Aggregation (X if more than one value exists) (NO2): A value is only shown in the result cell if all values entered into the result cell have the same value. In the case of standard aggregation NO2, the exception aggregation must also be NO2.

4. You can select an exception aggregation.
5. You can specify a reference characteristic for exception aggregation.

Currency/Unit

6. You can assign a fixed currency to the key figure. If this field is filled, the key figure bears this currency throughout BW. You can also assign a variable currency to the key figure.
7. You can specify an InfoObject that carries the unit of the key figure. For quantities or amount key figures, this field must be filled or you must enter a fixed currency or amount unit.

Properties

8. If you choose *High Precision*, the OLAP processor calculates internally with floating point numbers with 34 digits. This results in greater accuracy and fewer rounding differences. Normally, the OLAP processor calculates with floating point numbers with 16 digits.
9. If you select *Stock Coverage*, you can use a special key figure in a query to display how long the available quantity of a given stock value will cover the planned or expected demand for. More information: Stock Coverage
10. If you set the *Non-Cumulative* flag, the key figure becomes a non-cumulative key figure. You can then see and use the *Non-Cumulative* tab page. More information: Modeling Non-Cumulatives with Non-Cumulative Key Figures
11. If you select *Attribute Only*, the key figure that is created can only be used as an attribute for another characteristic. It cannot be used as a key figure in its own right in the InfoCube.

BI Clients

12. You can set the default scaling for the field in the query. This layout can be overwritten in queries.
13. You can set the default number of decimal places the field has in the query. This layout can be overwritten in queries.
14. You can make settings for *Read Access Logging (RAL)* . To include the key figure for *read access logging*, you must enter a *business area* and a *log domain* . You must also activate the property *Log Read Access Output* for the InfoProvider. More information:
15. Under *Properties* in the screen area under the Editor, you can see information about the key figure's DDIC elements. You can navigate further to the relevant information via the links.

> **Note**
>
> The generated view are only intended for internal use, not for use as an interface. You can use the external SAP HANA view instead, or official interfaces. More Information:

Related Information

Editing Non-Cumulative Key Figures [page 268]
Editing the Stock Coverage [page 268]

4.4.2.1 Editing Non-Cumulative Key Figures

Here you make further specifications for calculating non-cumulatives.

Context

The *Non-Cumulative* tab is available only if you have set the *Non-Cumulative* flag on the *General* tab.

Procedure

1. You can enter a key figure that represents the non-cumulative change of the non-cumulative value. There do not have to be values for this key figure in every time unit. For the non-cumulative key figure, values are only stored for selected times (markers). The values for the remaining times are calculated from the value in a marker and the intermediary non-cumulative changes.
2. Alternatively, you can specify two key figures that represent the inflow and outflow of a non-cumulative value.

> ⓘ Note
>
> For non-cumulatives with non-cumulative change, or inflow and outflow, the two key figures themselves are not allowed to be non-cumulative values, but must represent cumulative values. They must be the same type (for example, amount, quantity) as the non-cumulative value.

4.4.2.2 Editing the Stock Coverage

Here you specify the details for calculating the stock coverage.

Context

The *Stock Coverage* tab is available only if you have set the *Stock Coverage* flag on the *General* tab.

Procedure

1. You are in the BW Modeling tools. Go to the *Stock Coverage* tab page.
2. Under *Stock Type*, specify whether the referenced key figure for the stock is an opening stock or a closing stock.
3. Enter the maximum number of time periods that are taken into account. Note that the more time periods are specified here, the more the runtime will be affected.
4. Specify a referenced key figure for the stock. This can be either a non-cumulative key figure or a cumulative key figure.
5. Specify a referenced key figure for the demand.
6. Specify a time characteristic that specifies the granularity of the stock coverage. This characteristic also serves as the reference characteristic for the exception aggregation. An exception aggregation must always be specified for key figures for stock coverage.

4.4.3 Creating a Unit

Here you can create unit InfoObjects.

Procedure

1. You are in the BW Modeling tools. You have created a new unit using the wizard and you are in the editor.
2. You can still change the unit type that you selected in the wizard.
3. Under *Display* you can specify whether the text in executed query should be displayed, and if yes, which text. This setting can be overwritten in queries.

4.4.4 Creating XXL InfoObjects

You can create XXL InfoObjects here. These can store additional information for a characteristic as data type STRING or XSTRING.

Context

> **Note**
>
> Note that uploading any files carries the risk that a virus or other malware is distributed through this channel. Uploaded files are checked for viruses provided a virus scanner is connected. We strongly recommend that you configure a virus scanner.

Procedure

1. You are in the BW Modeling tools. You have created a new XXL InfoObject using the wizard and you are now in the editor.
2. You can still change the data type that you selected in the wizard.
3. For data type *Byte String*, you can still specify a *MIME Type*.

Open this video in a new window

4.4.5 Creating a Copy of an InfoObject

You can save an InfoObject under a different name in order to create a copy of it.

Procedure

1. You have opened an InfoObject in the editor. Choose *File* > *Save As ...* .
2. If required, select a different InfoArea to save the InfoObject under.
3. Give it a different name and a description.
4. Choose *Finish*. A copy of the InfoObject is created and is opened in the editor.

> **Note**
>
> To make sure that the texts are copied in all languages when saving the InfoObject under a different name, see SAP Note 2326441.

4.5 Creating InfoSources

You create an InfoSource to join two transformations.

Context

When InfoSources are created in the BW Modeling Tools, they are assigned to an InfoArea.

Procedure

1. You are in the BW Modeling tools. Open the context menu for your BW project here and choose *New* > *InfoSource*.

   As an alternative, you can create InfoSources from the data flow. To do this, go the *BW Object Types* in the editing screen for your data flow, select the type *InfoSource* and drag it to *Details*. Then open the context menu for this InfoSource and choose *Create Persistent BW Object*.

   You will be guided through the process by a wizard. More information: Creating InfoSources Using the Wizard [page 271]
2. The *General* tab opens.
3. If you set the *Aggregate Records Based on InfoSource Key Fields* flag, the incoming data will be aggregated with regard to the key.
4. If you set the *When filling the InfoSource, process 0RECORDMODE as if the target were an activated DataStore object* flag, InfoObject 0RECORDMODE will be added to the InfoSource. The aggregation behaviour is the same as for a DataStore object.
5. On the *Details* tab, select the InfoObjects that you want to be included in the InfoSource and/or create the fields.
6. Under *Manage Keys*, you can define keys for the InfoSource.
7. Activate the InfoSource.

Related Information

InfoSource [page 65]

4.5.1 Creating InfoSources Using the Wizard

Whenever you create an InfoSource, you are guided through the process by a wizard.

Procedure

1. 1. You are in the BW Modeling tools. In the context menu for your BW project, choose *New InfoSource*. You will be guided through the process by a wizard.
2. Select an InfoArea. To ensure that you can find your InfoSource again without difficulty, we recommend setting the *Add to Favorites* flag.
3. Enter a technical name and a description. The technical name can be up to 30 characters long.
4. By choosing *Copy from...*, you can select an InfoSource as a template. This allows you to copy over its properties to your new InfoSource and to edit them as required.
5. You can select a different object type as a template.

6. When you have finished making your settings, choose *Finish*. The editor appears.

4.5.2 Adding InfoObjects

You define the InfoSource by adding InfoObjects or fields.

Procedure

1. Choose the *Details* tab.
2. To add an InfoObject, choose *Add InfoObject*. Specify the start of the name, select an InfoObject from the result list, and press OK. You can also select multiple InfoObjects by pressing *Ctrl + Shift*. The InfoObjects are added to your InfoSource. Any compound characteristic are added too.
3. You can now configure further settings for the InfoObject properties in the right screen area. You can select an aggregation type under *Aggregation*.
4. Under *Manage Keys*, you can define keys.
5. To delete InfoObjects, choose *Delete*.

4.5.3 Adding Fields

You define the DataStore object by adding fields or InfoObjects.

Context

The advantage of using fields is that you do not need to model any InfoObjects. This is recommended especially if you want to load data types that are not supported by InfoObjects.

Procedure

1. Choose the *Details* tab.
2. To add a field, choose *Add Field*. The field is added to your InfoSource. You can now define the field in the right area of the screen.
3. Enter a description for the field.
4. You can specify an InfoObject under *Identify with:*. This changes the field into an InfoObject. The properties of the InfoObject are applied. You can undo this by switching to *Field* under *Identify with:*.
5. Enter the data type and data length.

6. Under *Manage Keys*, you can define keys.
7. To delete fields, choose *Delete*.

4.6 Creating DataStore Objects

The DataStore object is the central object for data storage and data consolidation in the BW system.

Procedure

1. You are in the BW Modeling tools. In the context menu for your BW project, choose *New* > *DataStore Object (advanced)*.

   You can also create DataStore objects from the data flow. To do this, go the *BW Object Types* in the editing screen for your data flow, select the type *DataStore Object (advanced)* and drag it to *Details*. Then open the context menu for this DataStore object and choose *Create Persistent BW Object*.

   You will now be guided through the creation process by a wizard. More Information: Creating DataStore Objects Using the Wizard [page 274]
2. The *General* tab opens. If you set the *External SAP HANA View* flag, an SAP HANA view is generated from the DataStore object. More Information:
3. Choose the properties that you need. More Information: Modeling Properties [page 71]
4. Under *Special Types*, you can select additional properties for the DataStore object. More Information: Modeling Properties [page 71]
5. Under *Data Tiering Optimization*, you can classify the data in your DataStore object (advanced) as hot, warm or cold, depending on how frequently it is accessed. More Information: and .
6. On the *Details* tab, select the InfoObjects that you want to be included in the DataStore object and/or create the fields. You can create groups in order to organize the fields and InfoObjects.

   More information:
   - Adding Fields [page 276]
   - Adding InfoObjects [page 275]
7. Optional expert setting: You can create partitions for key fields on the *Settings* tab.

   More Information: Creating Partitions [page 278]
8. Optional expert setting: You can create secondary indexes for a DataStore object on the *Settings* tab. The index is created on the active table. If no active table is found, the index is created on the inbound table. More Information: Creating Indexes [page 283]
9. Optional expert setting: On the *Settings* tab, you can prevent the creation of indexes that improve performance but that can also result in increased memory consumption. More information: Technical Settings [page 286]
10. Activate the DataStore object.
11. Under Properties in the screen area under the Editor, you can see information, about the DDIC elements for example. You can use the links provided to navigate to the relevant information.

> **Note**
>
> The generated views are only intended for internal use, not for use as an interface. You can use the external SAP HANA view or official interfaces instead.
>
> The external SQL view is an exception. This is released for customers. More Information: External SAP HANA SQL View [page 88]

Open this video in a new window

Results

The DataStore object can be integrated into a data flow.

Related Information

DataStore Object [page 70]

4.6.1 Creating DataStore Objects Using the Wizard

You will now be guided through the creation of a DataStore object by a wizard.

Procedure

1. You are in the BW Modeling tools. In the context menu for your BW project, choose *New DataStore Object*.
2. Select an InfoArea. To ensure that you can find your DataStore object again without difficulty, we recommend setting the *Add to Favorites* flag.
3. Enter a technical name and a description. The technical name can be between three and nine characters long. If you have a namespace for the DataStore object, the name can only be eight characters long.
4. You can select an object template.
5. When you have finished making your settings, choose *Finish*. The editor appears.

4.6.2 Adding InfoObjects

You define the DataStore object by adding InfoObjects and/or fields.

Procedure

1. Choose the *Details* tab.
2. To add an InfoObject, choose *Add InfoObject*. Specify the start of the name, select an InfoObject from the result list, and press *OK*. You can also select multiple InfoObjects by pressing *Ctrl + Shift*. The InfoObjects are added to your DataStore object. Any compound characteristic are added too.
3. You can now configure further settings for the InfoObject properties in the right screen area.

   Under *General*, you can use change type to change a field into an InfoObject and an InfoObject into a field. In this operation, the relevant field/InfoObject is deleted and a new one created. If the DataStore object already contains data, the data for this field/InfoObject will be lost. More information:

   Under *Criteria* you can specify which data can be loaded directly from a file into the DataStore object. More Information:
4. Under *Maintain Properties*, you can make aggregation and master data check settings for all InfoObjects/fields that support this:

   You can select an aggregation type under *Aggregation*.

   Under *Master Data Check*, you can decide whether you want to check for referential integrity or whether you want to run the check when data is loaded or queries are executed and whether you want to save the SIDs. Saving SIDs is useful if you have a compound characteristic with high cardinality and mainly navigation attributes of this compound characteristic are used in the query. This means the join must be executed very frequently, which can have a negative impact on performance. An additional column is created in the table for the purpose of saving the SIDs for this InfoObject. This column is filled during activation.

   > ⓘ Note
   >
   > InfoObjects of type display attributes do not have master data tables. It is therefore not possible to perform master data checks for these InfoObjects.
5. If you choose *Use Navigation Attributes in Extraction*, the navigation attributes of this InfoObject can be used in the transformation, if the DataStore object is the source of the transformation. In addition, you can use the navigation attributes as filter criteria in the DTP.
6. You can use groups to organize your InfoObjects. There are no semantic restrictions here. To do this, choose *Add Group*. You can change the technical name of the group and enter a description. You can move InfoObjects from one group to another using drag and drop.
7. You can define keys. More Information: Defining Keys [page 277]
8. To delete InfoObjects, choose *Remove*.

4.6.3 Adding Fields

You define the DataStore object by adding fields or InfoObjects.

Context

The advantage of using fields is that you do not need to model any InfoObjects. This is recommended especially if you want to load data types that are not supported by InfoObjects.

Note the restrictions that apply when modeling with fields: Modeling with Fields Instead of InfoObjects [page 39]

Procedure

1. Choose the *Details* tab.
2. To add a field, choose *Add Field*. The field is added to your DataStore object. You can now define the field in the right area of the screen.
3. Enter a description for the field.
4. You can now configure further settings for the InfoObject properties in the right screen area.

   Under *General*, you can use change type to change a field into an InfoObject and an InfoObject into a field. In this operation, the relevant field/InfoObject is deleted and a new one created. If the DataStore object already contains data, the data for this field/InfoObject will be lost. More information:

   Under *Criteria* you can specify which data can be loaded directly from a file into the DataStore object. More Information:
5. If you choose *Use Navigation Attributes in Extraction*, the navigation attributes of this InfoObject can be used in the transformation, if the DataStore object is the source of the transformation. In addition, you can use the navigation attributes as filter criteria in the DTP.
6. You can use fields to organize your InfoObjects. There are no semantic restrictions here. To do this, choose *Add Group*. You can change the technical name of the group and enter a description. You can move fields from one group to another using drag and drop.
7. You can define keys. More Information: Defining Keys [page 277]
8. To delete fields, choose *Remove*.
9. Under *Maintain Properties*, you can make aggregation and master data check settings for all InfoObjects/fields that support this:

   You can select an aggregation type under *Aggregation*.

   Under *Master Data Check*, you can decide whether you want to check for referential integrity or whether you want to run the check when data is loaded or queries are executed and whether you want to save the SIDs. Saving SIDs is useful if you have a compound characteristic with high cardinality and mainly navigation attributes of this compound characteristic are used in the query. This means the join must be executed very frequently, which can have a negative impact on performance. An additional column is created in the table for the purpose of saving the SIDs for this InfoObject. This column is filled during activation.

4.6.4 Defining Keys

You can define keys for the DataStore object.

Context

While you are defining the keys, the system only shows you fields and InfoObjects that are suitable for use as keys. The following fields/InfoObjects are therefore not displayed during when you define the keys:

- Fields of data type SSTR (short string), STRG (string) and RSTR (raw string)
- Only fields with undefined aggregation behaviour or aggregation behaviour *None*
- Key Figures

Procedure

1. You are on the Details tab. To define the keys, choose *Keys ...*. A dialog box appears.
2. Select the relevant InfoObjects and choose *Add to Keys*. The keys that you have just created are flagged with a grey key symbol.
3. You can move the InfoObjects in the list using the *Up* and *Down* buttons or by dragging them with your cursor.
4. Once you have finished defining your keys, press *OK*. The keys are then displayed on the General tab page.

4.6.5 Using Non-Cumulative Key Figures

You can use non-cumulative key figures in your DataStore object.

Prerequisites

The DataStore objects either has the properties

- *Activate/Compress Data* and *All Characteristics are Key, Reporting on Union of Inbound and Active Table* or
- *Activate/Compress Data* und *Write Change Log*.

The reference points that is required for calculation of the non-cumulatives is updated with the change log data. The DataStore object must therefore have a change log.

A key also must have been defined for the DataStore object. InfoObjects must be used as key figures, non-cumulative key figures and validity characteristics. Fields are not supported for this.

Procedure

1. If you select the *Inventory* flag on the *General* tab page, another tab page appears with the name *Inventory*.
2. You can now add a non-cumulative key figure. When the non-cumulative key figure is added, the key figure for non-cumulative changes, or the key figures for inflows and outflows, is/are also added to the DataStore object.
3. The time-reference characteristic is determined automatically if there is a valid time characteristic.
4. You can choose valid characteristics from the characteristics in the key, which are not time characteristics.

Related Information

Modeling Non-Cumulatives with Non-Cumulative Key Figures

4.6.6 Creating Partitions

You can create partitions for DataStore objects. The partitions can be created statically or dynamically.

Context

You can use partitioning to divide the entire dataset of a DataStore object into several smaller units that are independent and redundancy-free. Partitioning can improve system performance during data analysis. Partitioning can also be necessary if you are loading a large volume of data, because an unpartitioned table or a single partition can contain a maximum of two thousand million records. Partitioning is also needed if you use Data Tiering Optimization at partition level.

Partitions can be created for all DataStore objects with keys and for Data Mart DataStore objects. For DataStore objects with keys, you can use a key field as a partitioning characteristic. For Data Mart DataStore objects, every characteristic can be used as a partitioning characteristic.

No partitions can be created for staging DataStore objects with the property *Inbound Queue Only*. If you subsequently change the type to *Staging DataStore object* with the property *Inbound Queue Only*, all existing partitions are deleted.

Static Versus Dynamic Partitioning

Partitions can be defined statically, manually that is, or can be determined and created dynamically. You define static partitions as part of the metadata of the DataStore object at design time. Static partitions are thus transport-relevant, and changes to the partitions must be transported as metadata changes within the system landscape.

In the case of dynamic partitioning however, you only specify the partition field. The partitions are then created dynamically during activation of the data. If partitions do not contain any data, they are automatically deleted. Dynamic partitioning is therefore more flexible than static partitioning.

If you use dynamic partitioning without making any further settings, a new partition (EQUAL) is created for each new value of the partition field during data activation. This can result in a large number of partitions being created.

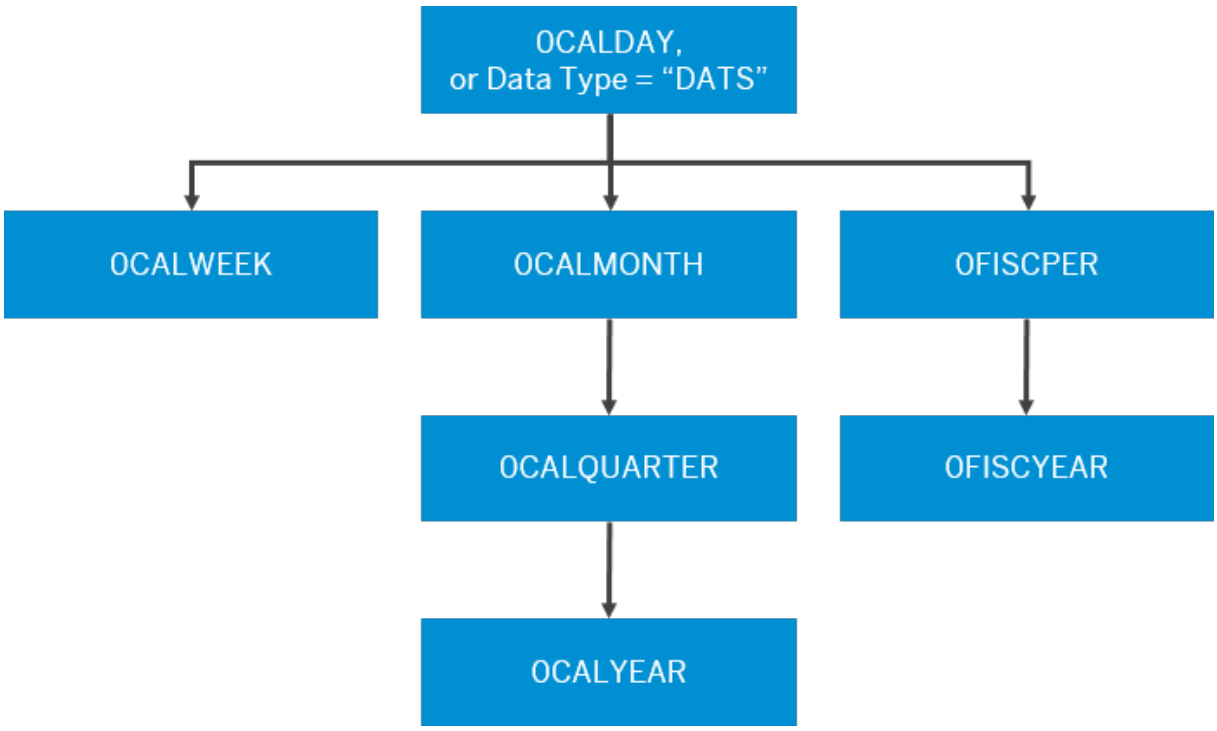
> ⚠ Caution
>
> Note that a large number of partitions has a negative effect on performance. There is also a limit to how many partitions are allowed. Dynamic partitioning is therefore only suitable for characteristics with low cardinality. If the maximum number of allowed partitions is exceeded for dynamic partitioning during activation of the data, the activation process terminates. In this case, you have to change the partitioning properties. You can select a different partitioning characteristic for example, or change the granularity.

If the partition field meets one of the following conditions, specifying a granularity will reduce the number of partitions to be generated:

- The partition field is an SAP time characteristic (0CAL*, 0FISC*) with total order.
- The partition field is a characteristic that is derived from one of the following characteristics: 0DATE, 0CALWEEK, 0CALMONTH, 0CALQUARTER, 0CALYEAR, 0FISCPER, or 0FISCYEAR.
- The partition field is a characteristic with data type DATS.

You can select Calendar Year (0CALYEAR) as the granularity for partition field Calendar Month (0CALMONTH) for example. During data activation, a new partition (with RANGE) is created for every value for which there is no corresponding RANGE partition yet. In the calendar year example, the corresponding calendar year is determined for every value of the partition field. If there is no RANGE partition yet for the calendar year, this is created, and all values that belong to this calendar year are summarized in the RANGE partition. Without granularity, a separate EQUAL partition would be created for every month appearing in the data instead. If the months 1.2019, 3.2019, 4.2020 and 6.2022 appear in the data for example, they are summarized in the following RANGE partitions: (1.2019 - 1.2020), (1.2020 - 1.2021) and (1.2022 - 1.2023).

Which values you can select for granularity depends on the partition field. The time hierarchy that defines which granularity is possible for which fields is shown below:



Note:

- If the partition field is 0CALDAY or a characteristic derived from 0DATE, or a characteristic with data type DATS, you can select 0CALDAY, 0CALWEEK, 0CALMONTH, 0CALQUARTER, 0CALYEAR, or 0FISCPER or 0FISCYEAR as the granularity.
- If the partition field is 0CALWEEK or a characteristic derived from 0CALWEEK, you can select 0CALWEEK as the granularity.
- If the partition field is 0CALMONTH or a characteristic derived from 0CALMONTH, you can select 0CALMONTH, 0CALQUARTER or 0CALYEAR as the granularity.
- If the partition field is 0CALQUARTER or a characteristic derived from 0CALQUARTER, you can select 0CALQUARTER or 0CALYEAR as the granularity.
- If the partition field is 0CALYEAR or a characteristic derived from 0CALYEAR, you can select 0CALYEAR as the granularity.
- If the partition field is 0FISCPER or a characteristic derived from 0FISCPER, you can select 0FISCPER or 0FISCYEAR as the granularity.
- If the partition field is 0FISCYEAR or a characteristic derived from 0FISCYEAR, you can select 0FISCYEAR as the granularity.

Dynamic Deletion of Empty Partitions

To allow automatic deletion of empty partitions during activation of the data, the following conditions must be met:

- You have chosen dynamic partitioning.
- There are partitions without data.

> ⚠ Caution
>
> Note that the system only considers a partition to be empty if all corresponding partitions of the various tables of the DataStore object are empty.
>
> The chosen temperature schema determines which tables have semantic partitioning and are thus relevant for dynamic deletion.

- The current temperature and the planned temperature must correspond to the default temperature. This is always the highest possible temperature (“hot” for example if hot and warm store are used, or “warm” if warm and cold store are used).

> ✤ Example
>
> In the case of hot/cold partitioning, only the active data (AT) table has semantic partitioning. If the AT partition is empty then, and there is no corresponding data in the cold store, the partition is removed even though the change log (CL) can contain entries with a key within the partition limits.
>
> In the case of hot/warm/cold partitioning, all tables - inbound queue (AQ), active data (AT), change log (CL), and the reference point table (RP) - have the same semantic partitioning. All corresponding partitions therefore have to be empty so that the corresponding dynamic partition is deleted during activation.

Procedure

1. Choose the *Settings* tab and expand the *Partitions* area.
2. Select a key field as the partition field. For Data Mart DataStore objects, every characteristic is available for selection.

   > ⓘ Note
   >
   > When selecting the partition field, note that the following characters are not allowed in the partitioning limits: ',', '$', '(', '-', and '*'
3. Select whether the partitions are to be created statically or dynamically.
4. If you want to define partitions **statically**, proceed as follows:
   a. You can create partitions by choosing *Add*. Specify a single value or a lower and an upper limit (interval) for your partition. Choose *OK and Next* to define a further partition. Choose *OK* to create the partitions. The partitions are numbered sequentially.

      > ⓘ Note
      >
      > Note:
      >
      > - The values of the individual partitions must not overlap each other.
      > - The upper limit is not contained in the partition. If you want to create seamless partitioning, the upper limit of a given partition must be identical to the lower limit of the next partition.

   b. Choose *Edit* to change the existing partitions.
   c. Choose *Split* to split the existing partitions. You can choose between *Simple Split* and *Equidistant Split*:
      - *Simple Split*: The partition is split into two partitions.

- *Equidistant split*: You have the option of specifying a time distribution using time characteristics (0CALDAY, 0CALMONTH, 0CALYEAR, 0CALWEEK, 0FISCYEAR, 0FISCVAR, 0QUARTER and their derived characteristics), and generate the partitions by day, month or year.

  To do this, you need BW Modeling Tools 1.19 or higher.

  The numbering of the partitions is automatically adjusted.

d. Choose *Merge* to merge existing partitions. Select two or more partitions and choose *Merge*.

e. You can delete partitions by choosing *Delete*.

5. If you have selected **dynamic** partitioning, the partitions are created dynamically during activation of the data.

   You can also select the required granularity for specific partition fields (see above) under *Partition Granularity*.

   Note the following with regard to data mart DataStore objects: For these objects, partitions are created already during loading or writing to the inbound queue (either via DTP or write API). This means that you can move the data to warm even if it has not been activated yet.

   > ⚠ Caution
   >
   > Moving to cold is only possible after data activation however.

6. If you have selected 0FISCPER or 0FISCYEAR as the granularity for dynamic partitioning, enter a constant for the fiscal variant in the DataStore object.

   a. Go to the *Details* tab.

   b. Select 0FISCVARNT from the field list.

   c. In the properties for 0FISCVARNT, choose *Criteria* > *Direct Input* and enter the required fiscal variant in the *Equals/From* field.

   d. Choose *Add*.

7. Activate the DataStore object.

   Static Partitioning: When the DataStore object is activated, the database table is created with all specified partitions.

   Dynamic Partitioning: When the DataStore object is activated, the table is created with just the overflow partition. If you have not specified a granularity, a partition is created (with EQUAL) during data activation for every value of the partition field. If you have specified a granularity, a new partition is created (with RANGE) during data activation for every value for which there is no corresponding RANGE partition.

   > ⓘ Note
   >
   > You can also change the following properties of dynamic partitioning for DataStore objects that already contain data:
   >
   > - The characteristic to be used for dynamic partitioning
   > - The granularity that has been set
   > - Both of these
   >
   > When this change is made, the system also creates a remodeling job.

   You can also **switch between** the two procedures (dynamic and static partitioning) at a later point in time. You can retain or change the partition characteristic. By switching in this way, you also create a remodeling job.

4.6.7 Creating Indexes

You can create secondary indexes for a DataStore object.

Context

> ⓘ Note
>
> This is an expert setting. There is usually no need to change this setting. Changing the indexes can be useful if a large number of individual records needs to be read in the DataStore object.

The index is created on the active table. If no active table is found, the index is created on the inbound table.

Procedure

Creating Indexes

1. Choose the *Settings* tab and expand the area *Indexes*.
2. Choose *New* to create a new index. The name of the index is generated automatically from the first available number that has not already been used for an index.
3. Select a field//InfoObject for the index and choose *OK*. The index is created and is displayed in the index area.
4. You can change or delete the index.

4.6.8 Define Anonymization Rules

To be able to anonymize personal data in the Data Protection Workbench, you first have to define anonymization rules for DataStore objects in the BW Modeling tools.

Prerequisites

In the general settings for the *BW Modeling* perspective, in the *Data Protection* area, you have set the indicator for *Maintenance of Anonymization Rules* (see Settings for BW Modeling Tools [page 37]).

This enables you to edit anonymization rules. The system adds the *Anonymization Rules* tab to the editor for editing your DataStore object.

Context

You can use the anonymization rules for DataStore objects to define which fields to change and to change them if a data record requires anonymization. You can use any field as the source for anonymization. To do this, you have to define a formula using selected SQL commands.

> ⓘ Note
>
> In most cases, anonymization of key fields of DataStore objects is **not** supported. You can only create anonymization rules for data mart DataStore objects only, even though all non-key figure fields are defined as key fields.

> ⚠ Caution
>
> Note that errors in the definition of a formula can cause errors during execution because the system only performs basic checks.

Procedure

1. Choose the *Anonymization Rules* tab.
2. From the list of fields on the left-hand side of the screen, select the field whose data that you need to anonymize after receiving an ILM notification for it.
3. Create the anonymization rule as a formula.

   > ✤ Example
   >
   > You want to replace the contents of the *Flight Passenger* field with the value *Anonymized* .

   The definition of anonymization rules is based on a special SQL syntax. For more details, see SQL Syntax for Anonymization Rules [page 285]. You use the Formula Editor in the BW Modeling tools for this. The simplest form of anonymization rule is setting a constant.
4. Test the anonymization you have defined by choosing the *Anonymization Preview* function from the toolbar in the BW Modeling tools.

   The system simulates your anonymization rules for a randomly selected number of records. It displays the result in an additional view, the *Anonymization Preview*.

Next Steps

Based on the anonymization rules you have defined, you can execute the *Anonymization* action in the Data Protection Workbench when working in your data protection worklist or your data protection templates.

Related Information

SQL Syntax for Anonymization Rules [page 285]

4.6.8.1 SQL Syntax for Anonymization Rules

The operators listed below are available for creating anonymization rules as formulas. A user-friendly formula editor is available in the BW Modeling tools.

Miscellaneous Functions

HASH_MD5	MAP	SYSUUID
HASH_SHA256	NEWUID	

String Functions

ABAP_ALPHANUM	LPAD	STRTOBIN
ABAP_NUMC	LTRIM	SUBSTR_AFTER
CONCAT	REPLACE	SUBSTR_BEFORE
LEFT	RIGHT	SUBSTRING
LENGTH	RPAD	
LOCATE	RTRIM	

Conversion Function

TO_BINARY

Date Function

CURRENT_DATE	CURRENT_UTCTIME	QUARTER
CURRENT_TIME	CURRENT_UTCTIMESTAMP	WEEK
CURRENT_TIMESTAMP	LAST_DAY	WEEKDAY
CURRENT_UTCDATE	MONTH	YEAR

More information

The functions listed here are a subset of the SQL functions in SAP HANA. For information about the syntax of the functions and examples of how they are implemented, we recommend the *SAP HANA SQL Reference Guide* (see SAP HANA SQL Reference Guide for SAP HANA Platform).

4.6.9 Technical Settings

You can make technical settings for a DataStore object.

> ⓘ Note
>
> This is an expert setting. There is usually no need to change this setting.

Memory Optimized Primary Key Structure

For DataStore objects with keys, the default setting is that the key is created as a concatenated index in the active data table (if the key consists of more than one element). This concatenated index provides optimized performance during access, but can also consume a lot of memory space. It is therefore possible to use an "inverted individual" SAP HANA index instead of a concatenated index. Although this requires significantly less memory space, it can negatively impact performance during access.

Suppress Index Structures for Compounded Characteristics

In the default setting, indexes are created for compound characteristics. This accelerates the join between the master data tables and the tables of the DataStore object during execution of a query. You can deactivate the option for creating these indexes if minimizing memory consumption is more important for you. It is advisable however to check whether this could a negative effect on performance.

Expected Number of Rows

Here you can specify the expected number of records:

- *Expected Number of Rows in largest Partition*: If there is semantic partitioning, you can specify the expected number of records in the InfoProvider's largest partition here.
- *Expected Number of Rows in largest Table*: If there is **no** semantic partitioning, you can specify the expected number of records in the InfoProvider's largest table here.

This information is used during creation and activation of a (empty) DataStore object in order to create an appropriate number of hash buckets on the SAP HANA database.

If this value is large, the system creates a large number of HASH partitions from the start. If it is small, a partition overflow might occur. The number of HASH partitions is determined from the specified number of data records and the current settings in SAP HANA table `TABLE_PLACEMENT`. It is still possible to adjust this value later on using the SAP HANA Landscape reorg mechanism (if an overflow threatens the 2 billion limit for a partition). For more information, see SAP Note 2374652.

> ⚠ Caution
>
> Specify a realistic value. If the value is greater than one billion (1,000,000,000), the system displays a warning.

This is on condition that the tables are first-level HASH-partitioned. The partitioning depends on the selected temperature schema. First-level HASH is used for the following temperature schemas: HO, WO, HWO, HCP, WCP. For more information, see SAP Note 2985173 - DTO: Data Tiering Optimization and Partitioning.

If a DataStore object already contains data, the value of this parameter is irrelevant. Especially during remodeling using shadow tables, the value from the older tables is copied.

> ⚠ Caution
>
> Create the value in your development system and transport it to your production system.

Merge Mode

You can change the default setting for DataStore objects here:

- *Smart Merge*: This is the default setting for all DataStore objects with the exception of DataStore objects that have a write interface. Whenever changes are made to the data of a DataStore object, the system triggers a *smart merge*.
- *Auto Merge*: This is the default setting for DataStore objects that have a write interface. Whenever changes are made to the data of a DataStore object as part of the write process, the system does **not** trigger a *smart merge* because a delta merge is executed for the activation in any case. This option is particularly useful in streaming and push scenarios and is therefore preset as the default setting for DataStore objects that have write interfaces.

  > ⓘ Note
  >
  > For more information, see DataStore Object with Write Interface [page 79] and .

4.7 Creating Semantic Groups

You can create semantic groups in order to create new components of the semantic group based on a reference structure. You can also include existing DataStore objects (advanced) in a semantic group.

Prerequisites

You are using an SAP HANA database.

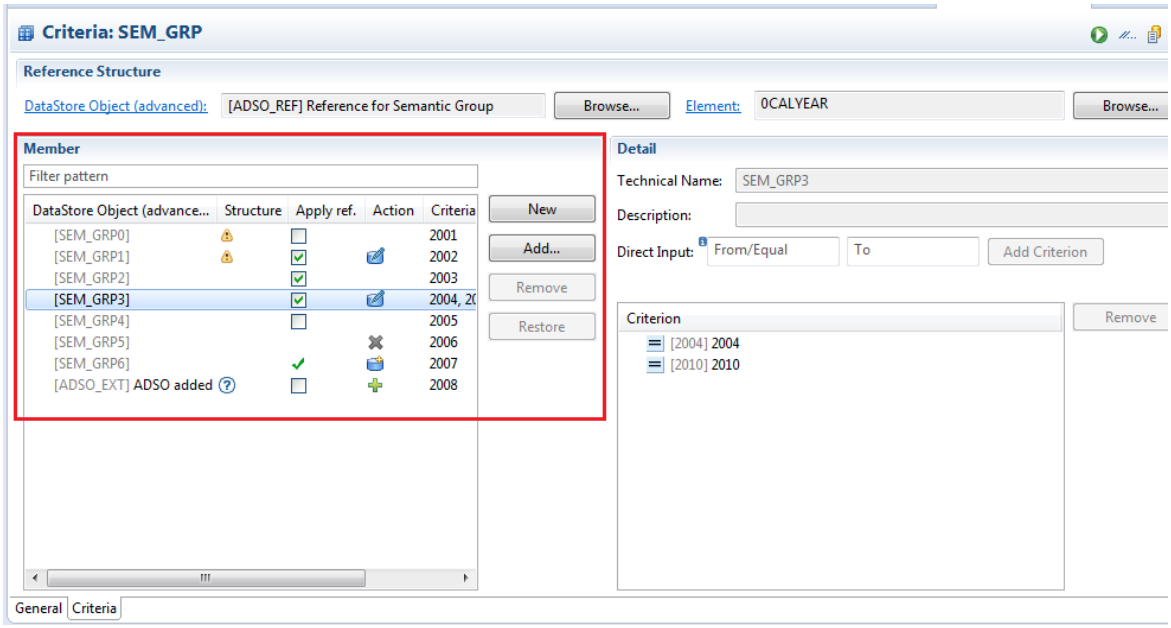
You have BW back end version 7.5 SPS04 or higher installed. You have installed and configured the BW Modeling tools.

Procedure

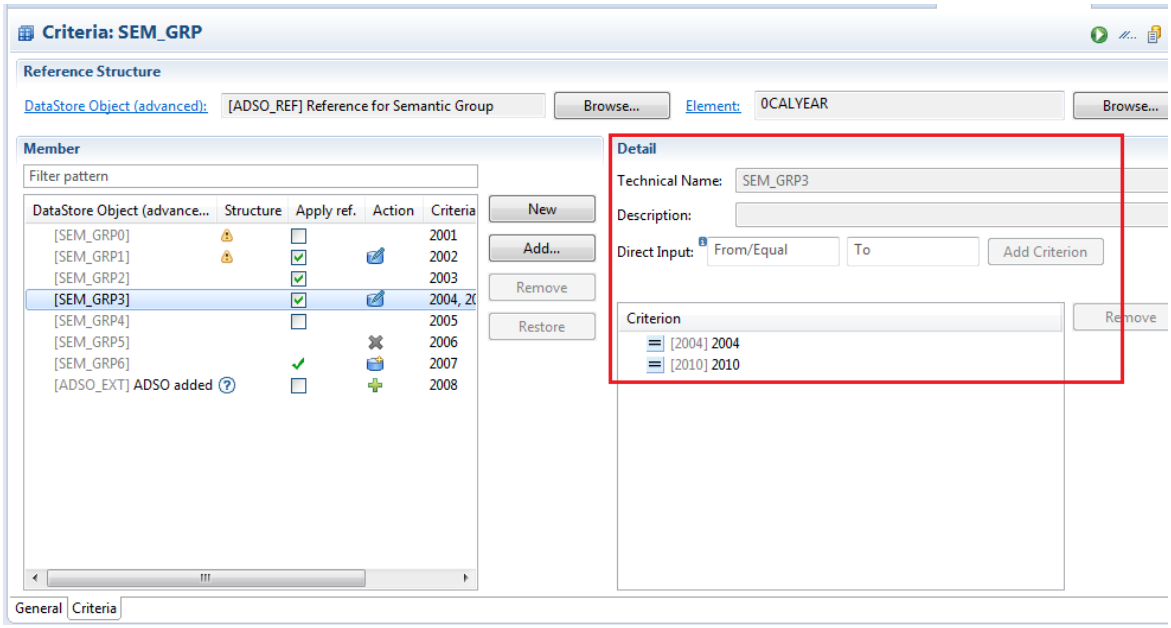
1. You are in the BW Modeling tools. In the context menu for your BW project, choose *New* > *Semantic Group*.
2. Enter a technical name and a description.

   > ⓘ Note
   >
   > Note that this character is not allowed in the technical name. This is because the forward slash is used for namespaces but the semantic group cannot be transported. However the objects generated by the group can be transported. Therefore namespaces can be used in the objects generated by the semantic group.

3. You can select an object template.
4. Choose *Finish*. The editor appears.
5. On the *General* tab, you can enter a name and description for a CompositeProvider to be generated. You should not enter the name of an existing CompositeProvider here. The components of the semantic group are then added to this CompositeProvider. It is always adapted whenever components of the semantic group are added or changed. You thus now have an object for analysis and reporting on the semantic group.
6. On the *Criteria* tab, select an existing DataStore object (advanced) as the reference structure. This serves as the structure template for the components of the semantic group.
7. Under *Element*, select a partitioning criterion. In the case of DataStore objects (advanced) with the property *Activate Data*, only elements are allowed that are in the key.
8. By choosing *New* under *Member*, you can create and add a new DataStore object (advanced) with the structure of the reference structure.



9. By choosing *Add*, you can add an existing DataStore object (advanced). If the DataStore object (advanced) is different to the reference structure, you can set the *Apply Reference Structure* flag. When the semantic group is generated, the structure of the DataStore object is then adapted. However you can also leave difference in the structure unchanged. Icons are displayed in the editor to indicate which action is performed when the semantic group is generated. More information: Example of the Definition of a Semantic Group [page 290]
10. In the *Detail* area under *Direct Input*, you can specify filter criteria. These can be individual values, ranges or combinations of these.



11. Choose *Start Generator*. The system displays a summary of the changes that generation would perform. Choose *Next*.
12. Specify a transport request for the transport to the production system. Choose *Generate*. The objects are generated.

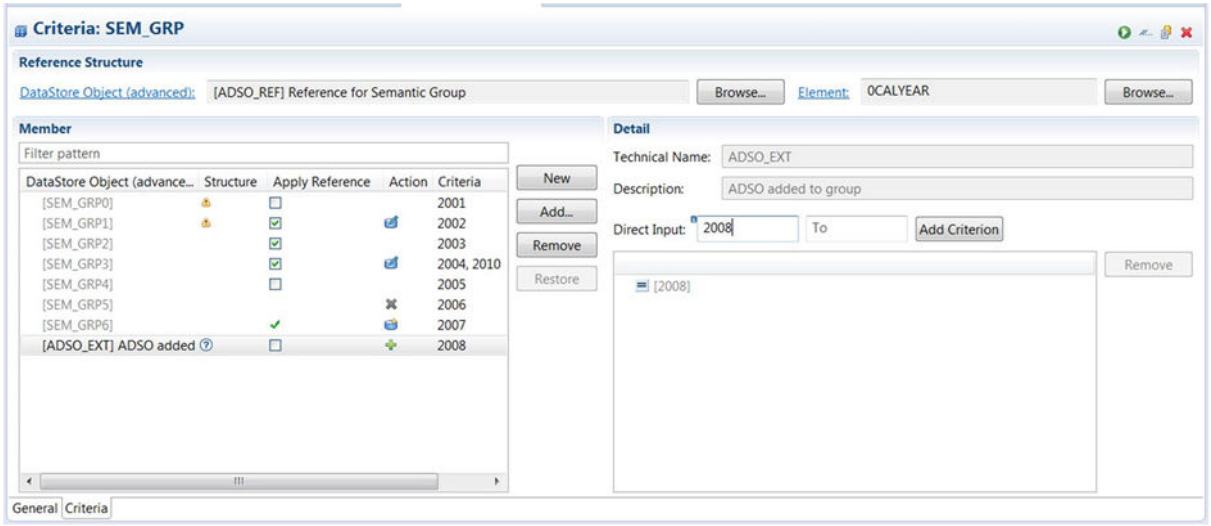
Related Information

Semantic Groups [page 91]

4.7.1 Example of the Definition of a Semantic Group

Icons are displayed in the editor to indicate which action is performed when the semantic group is generated.

When creating a semantic group, you can create new components of the semantic group based on a reference structure. You can also include existing DataStore objects (advanced) in a semantic group. These can have different structures. Depending on whether you choose the *Apply Reference Structure* option, and depending on whether the structure of a DataStore object differs from the reference structure, different icons are displayed in the editor. The icons show you which structures are adjusted during generation, which components have been changed and will be adjusted, which components will be deleted, and which will be added.

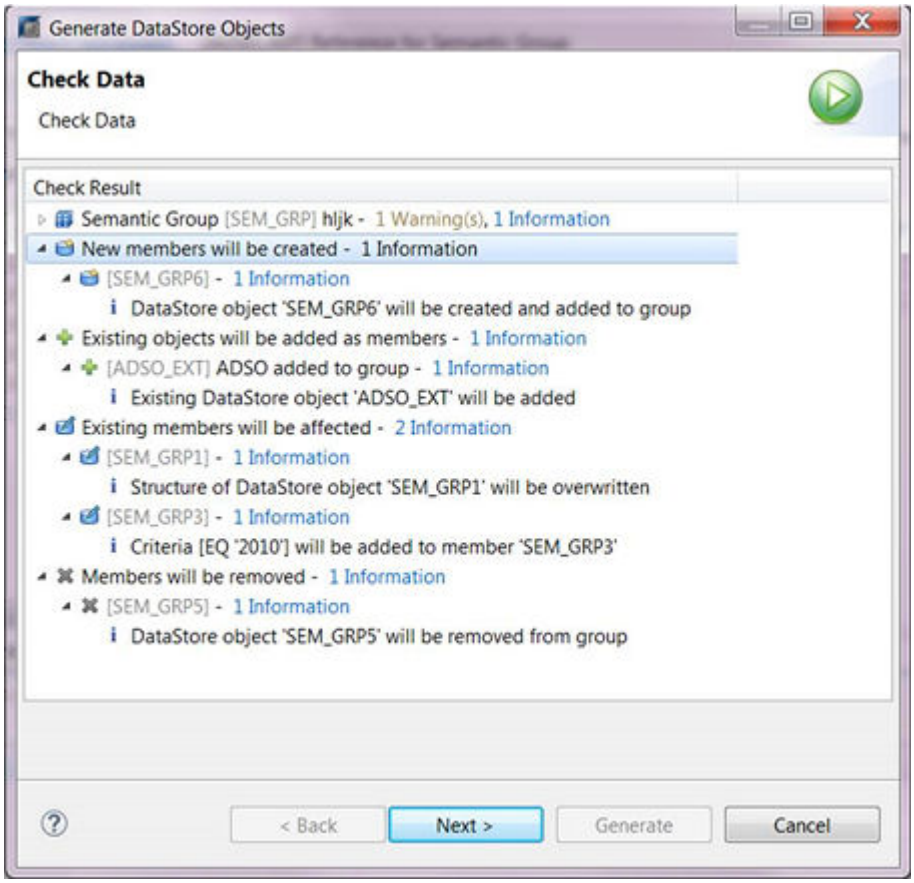


In this example, you see the following various statuses of DataStore objects (advanced):

- SEM_GRP0 is an active component whose structure is different to the reference. You can tell this from the exclamation mark on the icon. To prevent this structure from being overwritten, the *Apply Reference* flag is not set. With the next activation, there is therefore no action performed on this object.
- SEM_GRP1 is an active component whose structure is different to the reference. You can tell this from the exclamation mark on the icon under *Action*. As the *Apply Reference* flag is set, the structure is adjusted during the next activation. You can tell this from the Change icon.
- SEM_GRP2 is an active component whose structure matches the reference. As no criteria have been changed, no action is displayed. If changes were made to the reference structure however, these would be applied during the next generation, as the *Apply Reference* flag is set.

- SEM_GRP3 is an active component whose structure matches the reference. Changes have been made to the criteria here however, and the change icon is therefore displayed for the action. As with SEM_GRP2, future changes to the reference structure ar are applied, as the *Apply Reference* flag is set.
- SEM_GRP4 is an active component whose structure matches the template. As the *Apply Reference* flag is not set, future changes to the reference structure are not applied.
- SEM_GRP5 is an active component that is removed from the group during generation. You can tell this from the X under *Action*. The DataStore object (advanced) itself is retained.
- SEM_GRP6 is created again in accordance with the reference during the next generation and added to the group. You can tell this from the icon.
- ADSO_EXT is an existing DataStore object (advanced) that should be added to the group. At present it is not known whether the structure agrees or not. A question mark is therefore displayed. Next time it is saved, the corresponding icon should be displayed there. Under *Action*, a + is displayed.

If you choose *Generate* in this status, the system displays a list for each object showing which actions are performed on it, as in our example:



4.7.2 Changing the Semantic Group and Its Objects

You can change a semantic group, but note that restrictions apply.

Adding/deleting fields

You can add or delete elements (InfoObjects or fields) by changing the reference structure and then creating the group again. All components for which the flag *apply reference structure* is set are adjusted in accordance with the new structure. However, when deleting elements, please note that the reference characteristic must be retained.

Adding new components

You can add a new component to the semantic group at any time.

Deleting components

Active components can be deleted from the group. In this case, only the reference to the group is deleted from the metadata of the DataStore object (advanced); the DataStore object (advanced) itself is retained. If you want to delete the DataStore object (advanced), you need to execute this operation directly on the DataStore object (advanced).

Changing criteria

The criteria can be modified at any time without remodeling. If you restrict the criteria so that they no longer correspond to the loaded data, then remodeling is often necessary. For more information, see section *Remodeling after Generation* below.

Changing a DataStore object (advanced), which is the component of a group

After adding components, you can directly change the structure of individual components. You can make the change in the editing screen of the DataStore object (advanced) as usual. The editor displays information that the DataStore object (advanced) belongs to a group. Once this type of change is made, the structure of the DataStore object (advanced) generally deviates from the reference structure of group. Please note that the next activation of the semantic group will undo the change, if the *keep structure* flag is not set. When making changes, please note that the reference characteristic cannot be deleted from the DataStore object (advanced).

Differences between the development system/production system

You can only change semantic groups in the development system. In the production system, you can search for and display semantic groups, but cannot change them. When you display a semantic group, you only see its constituent parts with their criteria, and not see the reference structure or the CompositeProvider.

Remodeling after generation

If components of a semantic group already contain data, then remodeling may be necessary after generation. During the generation process, remodeling jobs are created for the affected components. These jobs need to be manually executed in transaction RSCNV_MONITOR. In this case, the generation of the semantic group runs without errors and only a warning is displayed for the affected components.

Remodeling is often necessary when criteria are restricted. If remodeling is not successful in this case, then you need to either adjust the criteria to match the loaded data or selectively delete the non-matching data from the DataStore object (advanced). Then you must restart the remodeling request.

4.7.3 Creating a Copy of a Semantic Group

You can save a semantic group under a different name in order to create a copy of it.

Context

When these are copied, a new group is created. This has the same number of partitions with the same criteria. All DataStore objects (advanced) are created again based on the reference structure. The names of the components are created automatically when copied, and can be changed before generating.

Procedure

1. You are in the BW Modeling tools. In the Project Explorer, choose *Copy* from the context menu on your semantic group.
2. Give the new object a name and a description.
3. Choose *Finish*. The new semantic group is opened in the editor and can be edited there.

4.7.4 Deleting Semantic Groups

You can delete a semantic group.

Context

If you delete a semantic group, the participating DataStore objects (advanced) and the CompositeProvider are retained. The references in the individual DataStore objects (advanced) to the group are deleted.

Procedure

You are in the BW Modeling tools. In the Project Explorer, choose *Delete* from the context menu on your semantic group.

4.7.5 Managing Semantic Group via API

You can use class `CL_RSO_SEGR_API` to manage components of a semantic group.

This API contains methods to create, read, update, or delete semantic groups.

Using this API, you can manage homogeneous semantic groups, in other words groups that contain DataStore objects that are identical copies of each other and that differ only in the criteria of the semantic group. You can use method `UPDATE` to change the layout of all components of a semantic group at the same time, as well as add new components or delete existing ones.

> ⚠ Caution
>
> Note that the API **does not** provide all of the features available in the BW Modeling tools. In particular, mixed semantic groups with DataStore objects that have different field lists, or DataStore objects with a different structure added using *Add*, are **not** supported.

> ⓘ Note
>
> For more information, see the documentation for class `CL_RSO_SEGR_API` in your system.

4.8 Creating CompositeProviders

In a CompositeProvider, you can merge data from BW InfoProviders with data from SAP HANA views using union and join, or just merge data from BW InfoProviders and SAP HANA views.

Prerequisites

You have installed and configured the BW Modeling tools.

If you want to use SAP HANA views that are located in HDI containers, you have to activate consumption of SAP HANA views in HDI containers under *SAP BW/4HANA Customizing Implementation Guide* > *SAP BW/4HANA* > *Data Modeling* > *Settings for Consumption of SAP HANA Views* in Customizing.

Procedure

1. Create the CompositeProvider. You are in the BW Modeling tools.

   You will now be guided through the creation process by a wizard. More Information: Creating CompositeProviders Using the Wizard [page 297]

   You can also create CompositeProviders from the data flow. To do this, go the *BW Object Types* in the editing screen for your data flow, select the type *CompositeProvider* and drag it to *Details*. Then open the context menu for this CompositeProvider and choose *Create Persistent BW Object*.
2. Define the properties for the CompositeProvider's runtime:
   - You can generate an SAP HANA view from the CompositeProvider. For more information, see the Help Portal under Generating SAP HANA Views from the BW System.
   - You can set the flag *This CompositeProvider can be added to another CompositeProvider*, in order to use the CompositeProvider as a provider in another CompositeProvider. More Information: Using a CompositeProvider as a Provider in Other CompositeProviders [page 297]
   - You can use the *Show Audit Characteristics in Query* flag to switch on the display of audit characteristics in the query. More Information: Audit [page 102]
   - If you activate the *Log Read Access Output*, then the CompositeProvider for the *Read Access Logging (RAL)* is included. Accesses to InfoObjects that are allowed are then logged. You can make this setting in the InfoObject editor. More Information:
   - Under *Common Runtime Properties* you can configure various settings at query runtime. More Information: General Runtime Properties [page 407]
   - *Runtime Profile Properties* contains the settings for processing the CompositeProvider. In most cases you can leave these settings unchanged. These are expert settings and can be changed later on if required. More Information: Runtime Profile Properties [page 412]
3. Select other participating InfoProviders or SAP HANA views and make assignments.

   More information:

   Creating Assignments [page 298]

4. You can create an aggregation or projection from the context menu or by pressing the corresponding speed button on a PartProvider. You can use an aggregation node to change the aggregation behavior of a key figure in the target structure.
5. You can define an SQL filter on a PartProvider. This feature should only be used by experts. To do this, choose *Create/Edit SQL Filter* from the context menu of a union, join, or projection node. The filter expression must be written in SAP HANA SQL. When you create the SQL filter, you receive a list of functions supported by SAP HANA.

   You can use a filter to restrict the selection to part of the data of the PartProviders in the CompositeProvider. To define a filter for a particular PartProvider, you have to create an aggregation node above the PartProvider. This is done at database level, which means that no conversion routines or currency shifts are performed when the function is applied.
6. You can define calculated fields in the target structure. To do this, you have to define an SQL expression in SAP HANA SQL. To do this, choose *Create Calculated Field* from the context menu of the target structure. When creating the SQL Expression, you receive a list of features supported by SAP HANA.

   This is done at database level, which means that no conversion routines or currency shifts are performed when the function is applied.
7. You can create fields in the target structure. To do this, choose *Create Field* from the context menu of the target structure. All characteristics have an additional *Forced Group by* flag in the properties. This modifies their icons. This function is available for all characteristics in the target structure, including calculated fields.

   These fields are not available in reporting; they are only visible in staging. Using calculated elements with an SQL expression however, you can convert the data type, so that it is suitable for reporting.
8. You can duplicate fields in the target structure. You can duplicate multiple fields at the same time.
9. Define the appearance of the CompositeProvider in the query.

   More information:

   Changing Properties for Fields of Type "Characteristic" [page 304]

   Changing Properties for Fields of Type "Key Figure" [page 307]
10. Activate the CompositeProvider.

Related Information

CompositeProvider [page 92]

4.8.1 Creating CompositeProviders Using the Wizard

Whenever you create a CompositeProvider, you are guided through the process by a wizard.

Procedure

1. You are in the BW Modeling tools. In the context menu for your BW project, choose *New* > *CompositeProvider*.
2. Select an InfoArea. To ensure that you can find your CompositeProvider again without difficulty, we recommend setting the *Add to Favorites* flag.
3. Enter a technical name and a description. A maximum of 12 characters can be entered for the technical name.
4. Choose *Next* or *Finish*:
   - If you choose *Finish*, the system automatically creates a CompositeProvider with union. The editor appears.
   - If you choose *Next*, you can initially select InfoProviders from the BW system or - if you have assigned a SAP HANA system to the BW project - select SAP HANA views that you want to be part of the union or join. If you do not select anything here, a CompositeProvider with union is created. If you want to create a CompositeProvider with join, you have to select either two providers in the join provider table or a join provider and multiple union providers. You can move the providers using *Drag & Drop* by choosing *Up* and *Down* from the context menu for the provider or by double-clicking on the provider.
5. Choose *Finish*. The CompositeProvider is generated. In the case of CompositeProviders with union and join, the union node is appended at the very bottom. The editor appears.

4.8.2 Using a CompositeProvider as a Provider in Other CompositeProviders

You are on the *General* tab. You can make the CompositeProvider available as a provider for other CompositeProviders by setting the *This CompositeProvider can be added to another CompositeProvider* flag.

This reusable CompositeProvider can then be used as the provider in another CompositeProvider.

This function is **not** available if

- If the CompositeProvider is already used in other CompositeProviders as a provider, you will see a where-used list and the relevant CompositeProviders are set to inactive when the CompositeProvider is activated.
- you want to add two or more reusable CompositeProviders in a CompositeProvider, which contain the same providers.
- a provider, which is part of the reusable CompositeProvider, is already contained in the CompositeProvider. You cannot use a provider more than once as a provider in a union operation.

> **Example**
>
> InfoProvider IC01 is used in CompositeProvider CP01, which is usable as a provider. CompositeProvider CP01 is added to CompositeProvider CP02. It is not possible to also add InfoProvider IC01 to CompositeProvider CP02.

- The CompositeProvider, which you want to use in another CompositeProvider, already contains a CompositeProvider.

There are also a number of scenarios that are not supported regardless. More Information: Scenarios for Using CompositeProviders in CompositeProviders [page 98]

4.8.3 Creating Assignments

You define the CompositeProvider by selecting other participating InfoProviders or SAP HANA views and creating assignments.

Procedure

1. You are on the *Scenario* tab page.
2. Select other participating InfoProviders or SAP HANA views.

   To add further objects to the union operation, you have the following options:

   - In the context menu of your uppermost node, choose *Add*. To find your provider, you can enter the technical name or description as search terms. This will only return providers that you can use in a CompositeProvider.
   - Press *(Add)*. Note that the speed button is not available for all node types.
   - You can use drag & drop to add BW InfoProviders from the Project Explorer to the scenario.
3. In the case of join assignments if the join providers are analytic indexes or SAP HANA views, you can change the cardinality by choosing *Edit Cardinality* in the context menu for a provider. For all providers, the cardinality is calculated first by the system. If the join condition or one of the providers is changed, the system checks the cardinality and changes it if required.

   > **Note**
   >
   > The cardinality setting can improve performance but can also produce incorrect data. Therefore in most cases you should not change the cardinality suggested by the system. An expert might need to change the suggested cardinality if the system does not recognize the data contents and s/he knows how to configure a more precise cardinality setting.

   The following change options are available:

System proposes	Possible change
n:m	1:m, n:1 or 1:1

System proposes	Possible change
1:n	1...1

4. In the case of join assignments, you can change the join type in the context menu. You can choose from the following join types: Inner Join, Referential Join (only 1:1 Join oder n:1 Join are allowed), Left Outer Join, Right Outer Join and Full Outer Join.
5. By pressing the *(Replace)* speed button, you can replace one provider with another.
6. In the area on the right of the screen, you can configure the assignments for the fields. On the left, you see the participating providers. To the right of these, the CompositeProvider's fields are displayed.

   The field types are distinguished by their icons:

   - Characteristic:
   - Characteristic with the *Forced Group by*: flag
   - Time characteristic:
   - Unit:
   - Key figure:
   - Calculated fields:

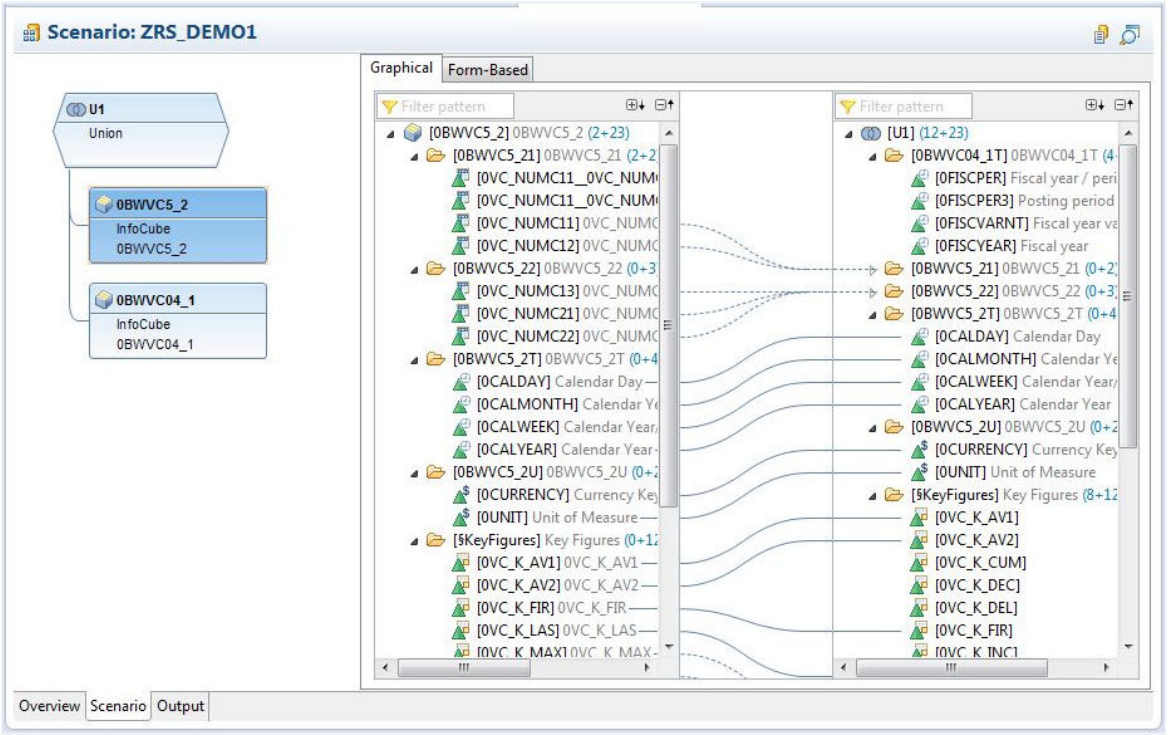
   In the graphical display, you can add the fields to the CompositeProvider either using drag and drop or by choosing *Create Assignments* in the context menu and create the assignments. If you add an InfoProvider by drag and drop, the fields and dimensions (groups) of the InfoProvider are applied and any required dimensions that do not already exist are created. You can also add complete dimensions by dragging the dimension to a free area or onto the root node. If you drag a dimension onto another dimension, only the fields are added to the new dimension.

   In the case of BW InfoProviders, the InfoObjects are added, and the dimensions are added as groups. If a field does not exist yet in the target structure, the system creates it. If navigation attributes have been added to the CompositeProvider, they are not navigation attributes any more.

   > ### Note
   >
   > When a CompositeProvider is opened again, only navigation attributes that have been assigned are displayed in the *Scenario* tab page. To see all navigation attributes of a participating InfoProvider, choose *Show Unassigned Navigation Attributes* in the InfoProvider's context menu. If you want to hide it again, choose *Hide Unassigned Navigation Attributes* in the context menu.

   If InfoObject names have been assigned to the fields of the SAP HANA view, these InfoObject names are used to create an association for the field in the CompositeProvider for SAP HANA views. When fields are assigned to the CompositeProvider, the associations are automatically set.

   If a SAP HANA view contains input parameters, these are displayed, and you can edit them like normal fields. You can only assign input parameters to other input parameters. An input parameter can be compounded either with an InfoObject or with another input parameter.



7. You can change the join operand under the root node of the target. To do this, choose *Change Join Operator* in the context menu of the join operand. If the join operand is *Equal To*, a double assignment is required. If the join operand is *Not Equal*, every column in the join condition must be assigned to a target field. The system proposes an assignment.

   The system performs this mapping automatically if not already done.

8. By choosing *Set PartProvider Runtime Properties* in the context menu of a participating InfoProvider, you can make the settings for the query properties. More information: Query Properties [page 101]
9. You can create groups in order to organize the fields of a CompositeProvider semantically. More information: Editing Groups [page 302]
10. By choosing *Check BW object*, you can check whether any assignments are missing.
11. In the context menu on one of the fields in your CompositeProvider, you can delete the assignments again.

4.8.4 SQL Expression

The filter expression must be written in SAP HANA SQL.

When creating it, you receive a list of features supported by SAP HANA.

This expression is executed at database level, which means that no conversion routines or currency shifts are performed when the function is applied.

4.8.5 Using Temporal Joins

You can use temporal joins in a CompositeProvider to depict time flows.

Prerequisites

The following restrictions apply:

- Only InfoObjects and DataStore objects are supported.
- Union nodes, aggregation, and projection are **not** allowed.
- SQL filters, SQL expressions, Force Group By, aggregation change and constant mapping are **not** allowed.
- The CompositeProvider cannot contain any fields that are not assigned.

Context

In SAP BW/4HANA, only master data can be defined as time-dependent data sources. Two additional fields/attributes 0DATEFROM and 0DATETO are added to the characteristic.

DataStore objects themselves cannot be defined as time-dependent however. However, they often contain time characteristics (from which a time interval can be derived), or a minimum of two InfoObjects that refer to 0DATE (which you can use to define a time interval for). This allows the DataStore objects in the CompositeProvider to be considered as time-dependent.

As soon as an InfoProvider that is contained in the CompositeProvider is made pseudo time-dependent, it is handled as a time-dependent data source. An important difference between pseudo time-dependent InfoProviders and time-dependent InfoProviders like InfoObjects, the system cannot prevent gaps or overlaps from occurring in the time stream. This always depends on the dataset of the pseudo time-dependent InfoProvider.

Procedure

Time-Dependent Characteristics

1. As soon as you use a time-dependent characteristic in the join, the join becomes time-dependent. You can also deactivate time dependency via *Hide Time-Dependent Fields*, provided that the characteristic does not exclusively consist of time dependent attributes. The time-dependent fields and the *Valid From* and *Valid To* fields are then hidden.

Time-Dependent InfoProviders

2. In the context menu of your InfoProvider, choose *Define Time Dependency*.

   You have the following options:

   - Select a time characteristic, which is contained in the InfoProvider to be processed as time-dependent and which can be used for time derivation (0CALMONTH, 0CALQUARTER, 0CALWEEK, 0CALYEAR, 0FISCPER, 0FISCYEAR). Time-dependency is indicated by the addition of the two date attributes

0DATEFROM and 0DATETO to the InfoProvider in the CompositeProvider (below the selected time characteristic).

> **Example**
>
> If 0CALYEAR is derived with the value 2014, the start date has the value 01.01.2004 and the end date has the value 31.12.2004

- Select a characteristic of type *Date* as the start date (*Valid From*) and another characteristic of type *Date* as the end date (*Valid To*). Make sure here that the dataset is suitable. The value of the attribute that is interpreted as the start date must be smaller than, or equal to, the value of the attribute that is interpreted as the end date. If this is not the case, the data record is interpreted as invalid, and the request will not be taken into account. Characteristics of type *Date* must have 0DATE as a reference.

3. You can also define a key date. To do this, choose *Set Key Date* in the context menu for your InfoProvider.

   Depending on the type of characteristic, there are various ways of defining a key date: Characteristics of type *Date* with a reference to 0DATE and the time characteristic 0CALDAY can be flagged as key dates. A key date derivation can be defined for time characteristics that describe an interval (0CALWEEK, 0CALMONTH, 0CALQUARTER, 0CALYEAR, 0FISCPER, 0FISCYEAR). You have multiple options:

   - Use the first day as the key date
   - Use the last day as the key date
   - Use a fixed day as key date (i.e. a particular day from the specified time period)
   - Key Date Derivation Type: You can specify a key date derivation type.

   An artificial characteristic CALDAY, which symbolizes the derived key date, is added to the InfoProvider, which a key date derivation is defined for.

4.8.6 Editing Groups

Groups are used to semantically structure the fields of a CompositeProvider. The group structuring is saved at InfoProvider level.

Procedure

1. You are on the *Scenario* tab page. In the context menu for the root node of your target or for a group, choose *Manage Groups* to call up the screen where you can edit groups.
2. By choosing *Create Group*, you can create new groups.
3. Enter a technical name and description for the group. Press *OK*. The group is created under the root node or under the group.
4. You can move fields between groups using drag and drop. You can use drag and drop to assign a field under a group.
5. You can form hierarchies for groups by assigning groups to be subgroups of another group using drag and drop. By pressing *ctrl + drag & drop*, you can change the order of the groups in the hierarchy.
6. Press *OK*. The group editing screen closes, and you return to the *Scenario* tab page.
7. By choosing *Move to Group* in the context menu for a field or a group, you can specify a group that the field or group should be made a child of.

8. Choose *Delete* in the context menu of a group to delete the group. All the fields in this group are moved to the group *Non-Grouped*.

4.8.7 Editing Properties

Context

On the *Scenario* tab, you can now edit certain properties of the CompositeProvider target fields. The *Output* tab provides more options for editing properties.

Procedure

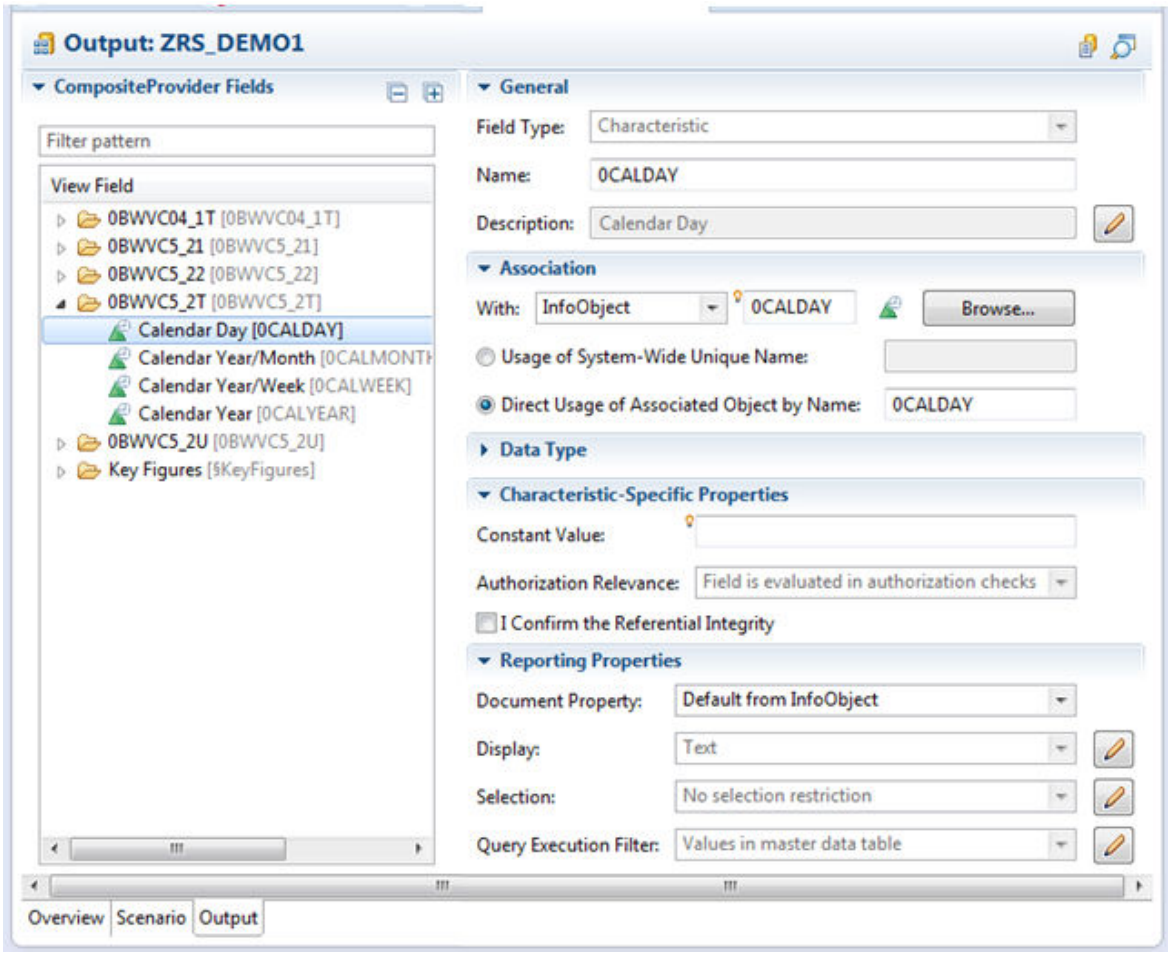
1. In the context menu on a field of type Characteristic in your CompositeProvider, you can assign a constant.
2. By choosing *Edit* in the context menu on one of the fields in your CompositeProvider, you can change the field's group assignment, type, name and description.

4.8.8 Changing Properties for Fields of Type "Characteristic"

You can change various properties for fields of type "characteristic" by changing the properties directly or by specifying a field association, which can pass on various properties to the field.

Procedure

1. You are in the *Output* tab. All fields in the CompositeProvider are displayed here. When you select a field, you can define or change various properties in the detail area on the right the screen.



2. Choose *Move to Group* in the field context menu to move fields between groups. Specify the new group in which you want to move the field. Choose *Browse* to open the dialog for editing groups. You can also move fields in the tree display by drag and drop.
3. By choosing *Association*, you can associate one of the following for a CompositeProvider field: an InfoObject of type Characteristic, Time or Unit, or an Open ODS view. We recommend that you use the association with InfoObjects if your BW system contains relevant InfoObjects. This means you can also use the master data of the InfoObject.

   The default properties and description of the associated InfoObject are applied. You can change these properties by choosing *Toggle Default Value/Manual Entry* and changing the value.

4. You can chose which name is displayed for this field in the query:

   - *Usage of System-Wide Unique Name*: A generated name is displayed in the query.
   - *Direct Usage of Associated Object by Name*: The info object name is displayed in the query.

   More Information: Names for CompositeProvider Fields in the Query [page 108]

5. If you have specified an association and you have set the *Direct Usage of Associated Object by Name* flag, you can activate the navigation attributes of the InfoObject or the Open ODS view. More Information: Activating Navigation Attributes [page 306]

6. If you have specified an association, you can set a constant from the InfoObject master data. The existing conversion routines are applied for the constant.

   If there is an assignment to this field for a provider, the constant functions as a filter. For providers without a link, the field is set to constant.

7. If there are no assignments or associations for a field, you can change the data type of the field.

8. Specify whether the field is authorization-relevant using *Authorization Relevance*. If the field does not have an association, or *Direct Usage of Associated Object by Name* is **not** selected, the field can be used in an authorization object. The local setting at field level is not affected by whether the field is used more than once. An authorization and the InfoProviders therefore decide whether the field is checked.

   You have the following options:

Authorization Relevance	**System Response**
The field has an association. You have also selected *Direct Usage of Associated Object by Name*.	The settings in the associated object are applied.
The field has an association. You have **not** selected *Direct Usage of Associated Object by Name* though.	You can choose whether the settings in the associated object are applied, or the local setting that determines whether the field should be included in the authorization checks.
The field has **no** association.	The local setting is applied.

9. For associated InfoObjects: If you set the *User confirmed Referential Integrity* flag, you have to ensure that only one master data record exists for each characteristic value in the provider. If referential integrity exists, this can improve performance. You can also configure this setting later on.

   > ### Note
   >
   > If you have set the flag for referential integrity, you have to make sure in future that only one master data record exists for each characteristic value in the provider. If you do not do this, you might find that records are missing in the query result, because, with referential integrity, the field values with the data of the associated InfoObject are assigned with an inner join.

10. You can define the following query properties under *Reporting Properties*:

    - *Document Property*:
      You can specify whether the document property is inherited from the InfoObject. You can then display any documents defined in the executed query.
    - *Display*:
      This property specifies whether individual characteristic values in the query are displayed, and how they are displayed:
      - No Display: The characteristic display is hidden. This function is particularly useful with the currency/unit characteristic for example, since the currencies are also displayed in the key figures.

- Key and Text: The characteristic values are displayed with their technical key and their text.
- Text and Key: The characteristic values are displayed with their text and their technical key.
- Text: The characteristic values are displayed with their text.
- Key: The characteristic values are displayed by means of their technical key.

Here you can select whether the Short Text, Long Text, or Medium Text should be displayed. You can specify which texts are available when editing the InfoObject.

- *Selection*:
  You can restrict the selection conditions.
- *Query Execution Filter*:
  This property specifies how filter values are selected for query execution:
  - *Values in master data table*:
    The system displays all characteristic values from the master data, independently of transaction data.
  - *Only posted values for Navigation*:
    The system displays posted values only.
  - *Only values in InfoProvider*:
    The system displays only values contained in the InfoProvider.

11. If you have selected *Direct Usage of Associated Object by Name* and have associated an InfoObject, you can activate navigation attributes on a field via the context menu provided that the navigation attributes on the associated InfoProviders have been activated too. Navigation attributes can be activated both at field level and as mass maintenance for each folder or if all fields are selected.

    An exception is made to this in the case of characteristics supplied by an InfoProvider that has a column view. The InfoProviders have one column view: DataStore object (with creation of SID values during activation), InfoObject In this case, navigation attributes can also be used even if they are not activated on the InfoProvider. An additional join is executed for the navigation attribute tables. This usually has a negative impact on performance. You should therefore always activate the relevant navigation attribute on the InfoProvider if possible.
12. Using the field context menu, you can specify compounding to one more other fields.

4.8.9 Activating Navigation Attributes

If you have specified an association, then you can activate the available navigation attributes.

Procedure

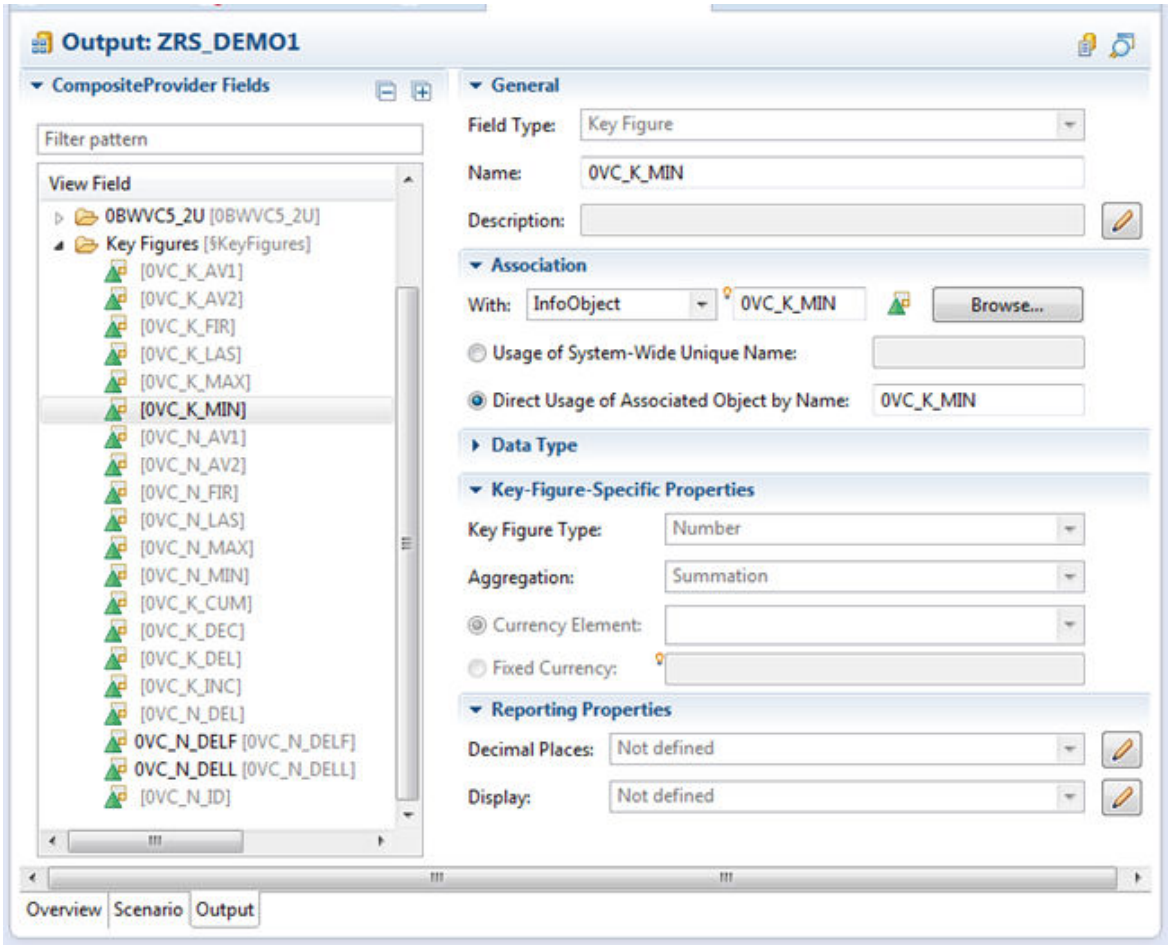
1. You are in the *Output* tab. Choose *Navigation Attributes* in the context menu on a CompositeProvider characteristic. This entry is only available in the context menu if navigation characteristics are available for the characteristic.
2. You see a list of the navigation characteristics. Select the navigation characteristics that you want to use.

4.8.10 Changing Properties for Fields of Type "Key Figure"

You can change various properties for fields of type "key figure" by changing the properties directly or by specifying a field association, which can pass on various properties to the field.

Procedure

1. You are in the *Output* tab. All fields in the CompositeProvider are displayed here. When you select a field, you can define various properties in the detail area on the right of the screen.



2. Choose *Move to Group* in the field context menu to move fields between groups. Specify the new group in which you want to move the field. Choose *Browse* to open the dialog for editing groups. You can also move fields in the tree display by drag and drop.
3. Under *Association*, you can associate a key figure for a CompositeProvider field. The default properties and description of the associated key figure are applied. You can change these properties by choosing *Toggle Default Value/Manual Entry* and changing the value.
4. You can choose which name is displayed for this field in the query:
   - *Usage of System-Wide Unique Name*: A generated name is displayed in the query.

- *Direct Usage of Associated Object by Name*: The info object name is displayed in the query.

More information: Names for CompositeProvider Fields in the Query [page 108]

5. If there are no assignments or associations for a field, you can change the data type of the field. The following data types are possible:

Type of key figure	Data type
Amount	CURR, FLTP, DEC
Quantity	QUAN , FLTP , DEC
Number	DEC , FLTP
Date	DEC , DATS
Time	DEC ,TIMS
Integer	INT4

6. In the case of an associated InfoObject, the aggregation behavior is adopted from this InfoObject. If you have not associated any InfoObjects, you can select a suitable aggregation behavior.
7. In the *Fixed Unit* field, master data from 0UNIT is displayed. In the *Fixed Currency* field, master data from 0CURRENCY is displayed.
8. The changeability of the *Key Figure-Specific Properties* has some dependencies:

Option	Description
Setting	**Changeability**
You have associated a key figure.	All properties are inherited from the associated key figure. The values under *Key Figure-Specific Properties* cannot be changed (with the exception of *Fixed Unit/Currency* and *Unit/Currency Element*).
You have associated a key figure and the *Fixed Unit/Currency* **flag is set in the associated key figure.**	The values under *Key Figure-Specific Properties* cannot be changed (including the *Fixed Unit/Currency* field).
You have selected *Usage of System-Wide Unique Name***.**	There may be more than one value under *Unit/Currency Element*.
You have selected *Direct Usage of Associated Object by Name***.**	The required *Unit/Currency Element* is set automatically when the matching element is assigned.

9. You can define the following query properties under *Reporting Properties*:
   - *Decimal Places*:
     This property specifies how many decimal places are used for the field in the query.
   - *Display*:
     This property specifies the scaling factor that is used to display key figure values in the query.

4.8.11 Changes to Used Objects

If you make changes to objects that are used in CompositeProviders, this can affect the CompositeProvider in some cases.

Deleting Objects Used in CompositeProviders

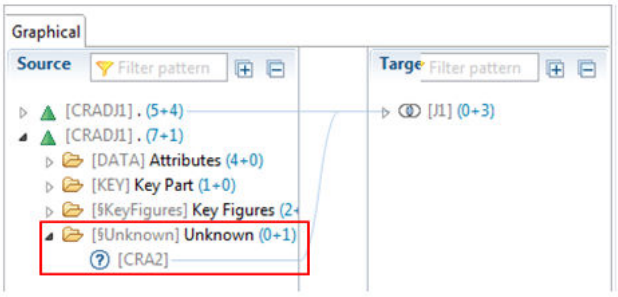
The following objects cannot be deleted if they are used in a CompositeProvider:

- InfoObject
- DataStore Object
- Aggregation Level

First remove the objects from the CompositeProvider and then delete them.

Changing Objects Used in CompositeProviders

If you delete InfoObjects from InfoProviders that are used in the CompositeProvider, then the CompositeProvider, which contains the corresponding InfoProviders, is set to inactive. The deleted InfoObjects are still displayed in the source. but are flagged as *Unknown* with a question mark.



Modify the affected CompositeProviders accordingly and activate them again.

Changes to InfoObject Properties

When certain changes are made to an InfoObject (compounding, time dependency, changing a display attribute to a navigation attribute and the other way round), the InfoProvider/CompositeProvider used is set to inactive. Modify the affected CompositeProviders accordingly and activate them again.

If you change navigation attributes to display attributes for example, this causes errors in the CompositeProvider runtime. You can resolve the errors in the CompositeProvider by removing the definition of the navigation attributes on the *Output* tab.

Changes to SAP HANA Views

When structural changes are made to SAP HANA views, the CompositeProvider remains active. However, errors occur in the CompositeProvider. You have to manually remove any deleted SAP HANA views, or deleted fields from SAP HANA views, from the CompositeProvider (*Target* area). The fields are displayed as *Unknown* with a question mark in the source. Using the BW object search in the BW modeling tools, you can find out the CompositeProviders in which specific SAP HANA views are used in. More Information: Searching for SAP BW/4HANA Objects [page 25]

4.8.12 Creating a Copy of a CompositeProvider

You can save a CompositeProvider under a different name in order to create a copy of it.

Procedure

1. You have opened a CompositeProvider in the editor. Choose *File* > *Save As ...* .
2. If required, select a different InfoArea to save the CompositeProvider under.
3. Give it a different name and a description.
4. Choose *Finish*. A copy of the CompositeProvider is created and is opened in the editor.

4.8.13 Supplying Data to 0INFOPROV

Every InfoProvider in a union is linked with the characteristic 0INFOPROV. This makes it possible to identify the involved provider, from which the data originates.

With a join, only one of the two sides of the join can be linked with the characteristic 0INFOPROV. With a CompositeProvider, only the left part is linked with 0INFOPROV.

Read requests for a CompositeProvider can only be split for providers that are linked with 0INFOPROV. Requests cannot be split for joins. This is important for functions that require a provider to explicitly specified. For InfoCubes with non-cumulative key figures, the InfoCube must be explicitly requested and must be linked with 0INFOPROV.

The 0INFOPROV link affects delta caching. Delta caching is only possible for providers that are linked with 0INFOPROV. For all other providers: when new data has been loaded, then all the data is read and not just the new data.

The 0INFOPROV link also influences whether the data integrity can be specified in the query definition. Setting data integrity is only possible for providers that are linked with 0INFOPROV. For all other providers: only data with the status *All InfoProviders up to released status (0)* is read.

Related Information

Delta Caching [page 418]

4.8.14 Processing Joins

When processing joins, remember that they are always executed in the SAP HANA database. There is no SQL implementation or ABAP implementation. For analysis in the Query Monitor, this means that you cannot toggle between execution in SAP HANA and execution in SQL (the *No optimized Operations in SAP HANA/BWA* option) if there are no joins involved.

You should also note that the data of every single InfoProvider must be aggregated before the join. You cannot prevent data aggregation before the join. If the cardinality of individual joins does not correspond to the data, this can result in key figure values being changed during navigation.

4.8.15 Settings for CompositeProviders

You can make a setting for the performance of the modeling environment in CompositeProviders.

By choosing *Window* > *Preferences* > *BW Modeling* > *CompositeProvider*, you can define starting from how many providers per CompositeProvider the system should switch to *Bulk Load*. The parameter only affects BW InfoProviders as participating providers.

In the default setting, the system switches to *Bulk Load* when a CompositeProvider contains at least 10 BW InfoProviders.

When a CompositeProvider with fewer than 10 providers is opened in the editor for the BW modeling tools, each provider is queried individually on the back end. These are then written to the cache and are thus quickly available next time the editor is opened. The information stored in the cache can also be used by other models without the need to access the back end again.

When a CompositeProvider with 10 or more providers is opened, the providers are not queried individually. The InfoProvider information is read as part of the CompositeProvider from the back end instead, and is not written to the cache. This allows the editor to open faster. When the editor is opened again however, the information must be read from the back end.

4.8.16 Pruning Queries on CompositeProviders

If a CompositeProvider contains several InfoProviders, it makes sense to use pruning for this CompositeProvider. Before going ahead with this, the system checks whether a participating InfoProvider (PartProvider) contains data for the selection area. If this is not the case for an InfoProvider, this InfoProvider is not relevant for this query and therefore does not need to be queried. To peform this check, the system uses the selection conditions for the query and the metadata of the InfoProvider.

There are various ways in which the restrictions for an InfoProvider can be stored:

1. Based on the definition of a constant
2. Based on the definition of a criterion for a characteristic
3. Based on user-specified restrictions

Based on the definition of a constant (not for DataStore objects)

If a characteristic is set as a constant, it can be compared with the selection range in the query. If the selection range does not contain the value of the constant, the system skips this InfoProvider when executing the query.

Based on the definition of a criterion for a characteristic (only for DataStore objects)

If you restrict a characteristic when defining a DataStore object, the system checks that the added data meets this restriction when loading into the InfoProvider. This ensures consistency at all time between the restriction in the metadata and the posted data.

If you add DataStore object like this to a CompositeProvider, the system saves the information about this restriction in the metadata (table `RSIPRORANGE`). When a query is executed on the CompositeProvider, the system checks prior to executing the query whether the selection area of the query and the area of the criterion for this InfoProvider overlap. If there is no overlap, the InfoProvider can be skipped during execution of the query.

Special case: Semantically partitioned objects

When a semantic group is generated, the system automatically writes the information about the partitioning characteristic as a criterion to the definition of the generated DataStore objects. This information cannot be changed in the DataStore object. If a CompositeProvider is also created in the semantic group, or a DataStore object that belongs to a semantic group is added to another CompositeProvider, the system stores this information in the metadata (table `RSIPRORANGE`) and uses it as described above for pruning during execution of the query.

Based on user-specified restrictions

You can use report `RSPP_PART_MAINTAIN` to directly edit restrictions for PartProviders of a CompositeProvider. The system stores this data in table `RSIPRORANGE` and analyzes it for pruning at query runtime.

This procedure can be used for all possible PartProviders of a CompositeProvider.

> ⚠ Caution
>
> Note however that the system does not check these conditions when loading. As a user, you need to ensure yourself that only data that meets the condition is loaded into the InfoProvider. Otherwise, a relevant InfoProvider might be ignored during execution of the query, in which case the result of the query will not be correct.
>
> For DataStore objects, we therefore recommend always using the specifications for the settings via the criteria.

Consideration of the Time Hierarchy

For DataStore objects with consistent time (with `TIME_IS_CONS` = X in table `RSOADSOLOC`) restrictions on a time characteristic can be converted into corresponding restrictions on a different time characteristic, thus allowing a comparison with the existing InfoProvider restrictions.

→ Tip

As an example, the system would thus convert a query restriction on `0CALYEAR` into a correspdonding restriction on `0CALMONTH` if there is a criterion on `0CALMONTH` for the DataStore object. The system can then perform the pruning check.

It is not possible to convert between all combinations of time characteristics however. Only those that can be quickly calculated are converted. In particular, it is not possible to convert between `0CAL` and `0FISC` time characteristics.

The following time characteristics are supported: `0CALDAY`, `0CALMONTH`, `0CALQUARTER`, `0CALYEAR`, `0FISCWEEK`, `0FISCPER`, `0FISCQRT` und `0FISCYEAR`.

Related Information

Manage Pruning for CompositeProviders [page 313]

4.8.16.1 Manage Pruning for CompositeProviders

Context

You can use program `RSPP_PART_MAINTAIN` to create or change entries in table `RSIPRORANGE`. These entries control which characteristics are used with which criteria for pruning.

Procedure

1. Call program `RSPP_PART_MAINTAIN` by choosing *Current Settings* > *Partitionspruning* in the navigation pane.
2. Specify your CompositeProvider.
3. You can have the system make a suggestion for each InfoProvider by choosing . This suggestion is based on the data that is already loaded. (Note however that this function is not available for all types of InfoProviders.)

   ⓘ Note

   You can also make entries manually. In this case, you need to ensure that the entries cover the data already loaded in the InfoProviders. The system checks this during activation. If this condition is not met, an error occurs.

You can use the operators EQ (equal to) and BT (between).

4. Choose  to copy the existing restrictions on an InfoProvider.
5. If new data is loaded, and the data no longer matches the ranges in this table, adjust the entries in the table. The  function helps you to do this.

   To avoid this problem, we recommend maintaining the criteria in the DataStore object definition. These criteria are then automatically transferred to table `RSIPRORANGE`. These conditions are also checked when loading into the InfoProvider however. The system can thus ensure that the pruning condition does not contradict the loaded data.
6. Activate your entries. Next time a query is executed on the CompositeProvider, these entries will be used for pruning.

4.9 Creating a Document Store

To be able to create and reuse documents in queries, you need a *document store*.

Context

When you create a document store, the system creates a DataStore object in the background as a physical document store. The name of this generated DataStore object is displayed in the maintenance transaction.

> ⓘ Note
>
> Make sure that you have activated the InfoObjects 0DOC* of the technical content, as the generated DataStore object requires this.

The fields of the DataStore Object are derived from the characteristics of the InfoProviders that are assigned to the document store. Since several fields are generated for each characteristic, SAP recommends choosing the list of assigned InfoProviders in such a way that the number of characteristics is not too large. Otherwise, the DataStore object cannot be generated any more.

> ⚠ Caution
>
> You can also find the generated DataStore objects whose name begins with `/1BW` using the search function in the BW modeling tools. Note that you **cannot** change the generated DataStore objects.

Procedure

1. You are in the *Project Explorer*. In the context menu for the required InfoArea, choose *New* > *Document Store*. The *New Document Store* dialog box appears.

2. Enter a name and a description for the document store.
3. If you want to use a document store as a template, choose *Browse* to select if from the existing document stores.
4. Choose *Finish*. The Document Store maintenance screen appears. On the *General* tab, the system displays the and the description.
5. To assign the document store to InfoProviders, switch to the *InfoProvider* tab and choose *Add*. The *Add InfoProvider* screen appears.
6. Specify the name of the required InfoProvider. You can also enter ***** to obtain a list of all InfoProviders. You can select one or more InfoProviders from this and choose *OK* .

   > ⚠ Caution
   >
   > The InfoProvider types supported for this are CompositeProviders and DataStore objects.

7. Activate the document store.

   > ⓘ Note
   >
   > Note that you have to activate the document store again if an assigned InfoProvider changes. The system then adjusts the generated DataStore object.

8. As an administrator, you can call **Document Administration** for the documents in this document store by pressing the spanner (wrench) symbol. There is also a function to download this overview.

   > ⓘ Note
   >
   > Note the authorization concept: You require the authorizations for standard authorization object `S_RS_DOCA` (activities `DISPLAY` and `MAINTAIN`).

   If you have selected a row, you can perform the following functions:

   - By pressing *Details*, you can call a display of the details for the selected document, including header data (dokument ID, version), metadata (docment type and document status, user name, key figure name, InfoProvider that belongs to the query with which the document was created) and further technical data.
   - By choosing *Show Document*, you can call a preview of the document.
   - By choosing *Show SELDR*, you can view the filter definition of the cell. You can check the selection conditions here.
   - By choosing *Delete Keys*, you can delete the document (and all existing versions). (The keys are determined from the filter in the cell and serve to rapidlay assign the documents to a query.)

     > ⓘ Note
     >
     > If you want to delete multiple documents at once, you can use report **RSDOC_DELETE_TAGGED_DOCS**.
     >
     > You can use report **RSDOC_DELETE_DOCUMENTS** to physically remove deleted documents from the database.

Related Information

Document in the Document Store [page 110]
Document Support [page 443]

4.10 Working with Queries

You define a query in order to analyze data in the SAP BW/4HANA.

Prerequisites

You have created a BW project in the BW Modeling perspective and connected it with a back end system.

Context

The main parts of the query definition are navigation and filters:

- For **navigation**, you select free characteristics and define the content of the rows and columns in the query. You use this selection to specify which data areas of the InfoProvider you want to navigate through. The alignment of row and column content determines the initial view for the query.
- The selections in the **filter** apply to the query in its entirety. When defining the filter, you select characteristic values from one or more characteristics. You can also set a key figure as a filter. If you do this, the query only has characteristics in the rows and columns. All the InfoProvider data is aggregated using the filter selection for the query.

You can parameterize queries by using variables for characteristic values, hierarchies, hierarchy nodes, formulas and texts, or by defining new variables where necessary.

To make the selection of InfoObjects more precise, you have the following possibilities:

- You can restrict characteristics to characteristic values or hierarchy nodes.
- You can define **selections**.
- You can define **formulas**.
- You can define **Exception Cells** and **Help Cells**.
- You can use **local structures**.
- You can define **conditions**.
- You can define **exceptions**.
- You can define **calculated and restricted key figures** as reusable query components.

Procedure

1. Create a query using the wizard.
2. Define the query properties in the editor.

Related Information

Creating a Query Using the Wizard [page 317]
Defining Queries in the Editor [page 318]
Opening and Changing Existing Queries [page 444]
These links take you to the documentation for modeling queries.
Query [page 111]
This link takes to you to the documentation for query concepts.

4.10.1 Creating a Query Using the Wizard

Whenever you create a query, you are guided through the process by a wizard.

Context

In the Project Explorer view, you can create a query either for a BW project or for an InfoProvider. You can also create a query as a copy of an existing query.

Procedure

1. To do this, open the context menu for a BW project or an InfoProvider and choose *New* > *BW Query*, or open the context menu for the query you want to use as a template and choose *Copy To ...*. The *New BW Query* dialog box appears. Depending on your point of entry, the system will have already preselected certain objects.
2. The system preselects the current BW project at the very least. By choosing *Browse*, you have the option of selecting a different BW project.
3. If you have created a query for a BW project, you also have to select the required InfoProvider. Enter its technical name or a corresponding search term. The system helps you when searching for the technical name, and returns a list of results under *Matching items*.
4. If you have created a query for an InfoProvider, the system preselects this InfoProvider. By choosing *Browse*, you have the option of selecting a different InfoProvider.

5. To add your query to your local favourites, choose *Add to Favorites*.
6. Enter a unique technical name and a description for your query.
7. If you have created a query as a copy of an existing query, the system preselects this query. By choosing *Browse*, you have the option of selecting a query as the template.
8. Choose *Finish*.

Results

The system opens the new query in the Eclipse Editor. The *General* tab opens, together with all query settings that you have made so far.

Related Information

Defining Queries in the Editor [page 318]
This link takes to you to the documentation for modeling queries.
Query [page 111]
This link takes to you to the documentation for query concepts.

4.10.2 Defining Queries in the Editor

You can define the properties of a query in the native Eclipse editor.

Context

On the *General* tab you will find all the query settings that you have made so far.

Procedure

1. Choose the *Sheet Definition* tab.
2. Press the *Show InfoProvider View* button.

   The available objects for the InfoProvider you selected are displayed as a directory tree in the InfoProvider view:
   - *Key Figures*
   - *Characteristics*

- Reusable objects *Reusable Components*): *Calculated Key Figures*, *Restricted Key Figures*, filters and structures.

> **Note**
>
> You can drag the InfoProvider View to another location of your choosing.

3. Specify which characteristics, key figures, or structures you want to be available for navigating through the query data.
   a. In the *BW Repository*, choose the required characteristics and drag them to the relevant part of the query:
   b. Define the required selections - with and without key figures.

   If you select an InfoObject, you can view and edit its *properties* on the right of the *Sheet Definition* tab.
4. Define the required formulas, filters conditions and exceptions.
5. Save the query definition.

Results

To view the query in the preview, choose *File* > *Refresh* and then *Show Preview*.

Related Information

Including Characteristics [page 320]
Defining the Structure [page 327]
Defining Structure Elements [page 328]
Defining Exception Cells and Help Cells [page 358]
Defining Priorities [page 360]
Defining Filters [page 362]
Defining Conditions [page 367]
Defining Exceptions [page 370]
Saving Queries [page 374]
Using Reusable Query Components [page 379]
Query Properties (General Tab) [page 397]
Dependency Structure Tab [page 405]
Changing Query Runtime Properties (Runtime Properties tab) [page 406]
These links take you to the documentation for modeling queries.
Query [page 111]
This link takes to you to the documentation for query concepts.

4.10.2.1 Including Characteristics

By including a characteristic in the query, you specify that this characteristic should be available for navigation in the query data.

Procedure

In the *BW Repository*, choose the required characteristics and drag them to the relevant part of the query: In the *Rows*, *Columns* or under *Free Characteristics*. Alternatively, you can choose *Add Characteristics* in the context menu. The dialog box for selecting characteristics appears.

If you insert a characteristics that has a compound characteristic, the system automatically inserts this too. This also applies to copying and pasting. The system always inserts the compound characteristic into the filter's characteristic restrictions.

> ### Example
>
> Characteristic Country is compounded with characteristic Region. Drag characteristic Region to Free Characteristics. The system automatically inserts characteristic Country into the filter's characteristic restrictions.

You can use as many characteristics as you want in the query.

> ### Restriction
>
> If you want to release the query for *Remote Access by OLE DB for OLAP* (in the properties of the query, Screen Area: Remote Access [page 400]), note the following restriction: Rows and columns together can contain a maximum of 50 - <number of structures> characterstics.
>
> If the query does **not** contain a structure, a total of 50 characteristics are permitted in the rows and columns. If the query contains a structure, a total of 49 characteristics are permitted in the rows and columns. If the query contains two structures, a total of 48 characteristics are permitted in the rows and columns.

Related Information

Characteristic [page 112]
Properties of the Characteristic [page 321]
Tab: General [page 321]
Tab: Hierarchy [page 325]

4.10.2.1.1 Properties of the Characteristic

If you have inserted a characteristic in the query definition, the properties of this characteristic appear in the screen area on the right.

The *Properties* screen area is divided into tab pages, each of which provides different settings.

> ⚠ Caution
>
> Note that you can activate manual input for the settings by using ✎.
>
> If you want to use a default value specified on the object again, choose ✎. The system displays the default settings as non-input-ready fields.

Related Information

Tab: General [page 321]
Tab: Hierarchy [page 325]

4.10.2.1.1.1 Tab: General

On the *General* tab, you can view the properties of the characteristic and make various settings.

General

This area displays the technical name and the description.

You can change the description. The new description text is different to the text in InfoObject maintenance and is used in the query display.

Value Output Format

Under Value Output, the system displays the output type. To change this, choose ✎.

Under *Display As*, you can specify whether the values of the characteristic are displayed, and how. You can choose here from the options created for the characteristic in InfoObject maintenance. The following table provides an overview of the possible display options in the BW system:

Display Type

Display Type	Description
Do Not Display	No display: The characteristic is hidden. This function is particularly useful with the currency/unit characteristic for example, since the currencies are also displayed in the key figures.
[Key] Text	Key and Text: The characteristic values are displayed by means of their technical key and their description text.
Text	Text: The characteristic values are displayed by means of their description text.
[Key]	Key: The characteristic values are displayed by means of their technical key.
Text [Key]	Text and Key: The characteristic values are displayed by means of their description text and their technical key.

If you set a display type that contains text, you can select a particular text type under *Text Output Format*. You have the following possibilities:

Text Display

Display Type	Description
Standard	Standard: The shortest available text is used for the characteristic values.
Short	Short text: The short text for the characteristic values is used.
Medium	Medium text: The medium text for the characteristic values is used.
Long	Long text: The long text for the characteristic values is used.

Sorting

Under Sorting, the system displays the sorting criteria. To change this, choose .

Sorting Criteria

Criterion	Description
Sorting Attribute	Sorting Attribute: Select either this characteristic or the attribute that you want to sort by. → Tip You want to sort your articles according to specific criteria. To do this, you create an attribute for the Article characteristic that contains these criteria, and sort the articles by this attribute.
Sort by	Sort by: Text or Key
Sort Direction	Sort Direction: Ascending or descending

Result Output Format

Here, you can decide under which conditions and in what form you want result rows to be displayed:

Result Row Display

Option	Description
Always	Always: The result rows are always displayed.
Only if more than one child	If there is more than one individual value: The result rows are displayed if there are two or more individual values.
Never	Never: The result rows are always suppressed.

Cumulate Values: You can specify for each characteristic that the key figure values can be cumulated for the relevant characteristic attributes - regardless of the aggregation behavior of the key figure.

Example

- If the key figure has the standard aggregation SUM, the individual values of an area are cumulated; the first value is added to the second value, the result is added to the third value, and so on.
- If the key figure has the standard aggregation MIN, the system checks two consecutive values to determine which value is smaller. The systems this value as the cumulative value and compares it to the third single value.

In the columns, the values are cumulated from top to bottom, and in the rows, the values are cumulated from left to right. With blocks of single values, a drilldown in both the rows and the columns, the values are cumulated from top to bottom and from left to right.

Caution

Note that you can only cumulate with characteristic values and not with the structural elements of a characteristic structure. The *Cumulate Values* function only takes effect if you have a characteristic instead of a characteristic structure in the drilldown. If you use two structures however - the key figure structure and a characteristic structure - you can cumulate using cell-specific definitions.

Display Level

This allows you to categorize a characteristic in the filter. This means that you can categorize purely technically filters (which are **not** relevant for end users) as the detail overview (*In Detailed Overview Only*). These are then only displayed if the detailed view is set for display in the front end. You can therefore create a special view of the filtered characteristics that is tailored to end users.

The following categorizations are available:

- *Also in Simple Overview* The characteristic always appears in the list of static filters
- *In Normal Overview* The characteristic only appears in the list of static filters if the (*In Normal Overview*) or (*In Detailed Overview Only*) option is selected.
- *In Detailed Overview Only* The characteristic only appears in the list of static filters if the (*In Detailed Overview Only*) option is selected.

Extended

Under *Access Type for Result Values*, you can define the access type for result values. You have the following options:

Access Type for Result Values

Option	Description
Master Data	Master data: When the query is executed, a row of data is always displayed for the characteristic. As long as values have been maintained for this characteristic in the master data, they will be displayed here. The display does not depend on transaction data.
Posted Values	Posted values: When the query is executed, a row of data is only displayed for the characteristic if there are posted values for this characteristic.
Characteristic Relationships	Characteristic Relationships: When the query is executed, a row of data is only displayed for the characteristic if you have modeled corresponding characteristic relationships.

Under *Filter Value Selection*, you can define how filter values are selected when the query is executed. You have the following options:

Filter Value Selection During Query Execution

Option	Description
Only Values in InfoProvider	Only values in the InfoProvider: The system only displays values contained in the InfoProvider.
Values in Master Data Table	Values in the master data table: The system displays all characteristic values from the master data, independently of transaction data.
Only Posted Values for Navigation	Only posted values for navigation: The system displays posted values only.

Under *Refresh Variables*, you can define how to refresh the variables that you use in default values or for defining presentation hierarchies. You have the following options:

Refresh Variables

Option	Description
Dynamically	Refresh dynamically: If you choose this option, the system refreshes the application using the settings from the current navigation view.
As Designed	Refresh in accordance with the design: If you choose this option, the system refreshes the application using the hierarchy and filter settings in the query definition.

Under *Value Help in New Lines*, you can specify whether the value help for the characteristic value in new rows is according to master data or characteristic relationships.

4.10.2.1.1.2 Tab: Hierarchy

If there are one or more hierarchies for the selected characteristic, you can select the required hierarchy and view or change its properties by going to the *Hierarchy* tab.

General

In this screen area, you can select the hierarchy that you want to use in the query (*Activate Hierarchy*). You can also deactivate the hierarchy, select a different hierarchy for the selected characteristic, and view or change the name and the date of the hierarchy. You can also see the version of the current hierarchy.

Display of Nodes

Each hierarchy has properties that you specify in the hierarchy definition. To change the properties for the current hierarchy, choose .

Here you can override the following hierarchy properties for the hierarchy node display:

Display of Hierarchy Nodes

Option	Description
Position of Child Nodes	Position of child (lower-level) nodes: Choose whether you want to position the lower-level nodes either (*above*) or (*below*). Choose "above" if you want to see the overall result at the bottom of the screen, for example.
Nodes with Only One Child Node	Nodes with only one child (lower-level) node: Choose whether you want to show (*Always Show*) or hide (*Hide*) nodes with only one child node.
Values of Postable Nodes	Values of posted nodes: Choose whether you want to show (*Always Show*) or hide (*Hide*) the values of posted nodes. In hierarchies, nodes always display the aggregated values of the lower-level nodes and leaves. To display this value posted to the node, the system inserts an additional leaf with the same name as the node. This leaf also displays the value posted to the node.
Expand to Level	Expand to level: Choose the hierarchy level to display the hierarchy to when expanded.

Sorting

In the hierarchy, you can set the sorting according to the key (*Key*) or text (*Text*) in ascending (*Ascending*) or descending (*Descending*) order. To change the sort criterion or the sort order for the current hierarchy, choose .

4.10.2.2 Defining the Structure

You can use structures both in the *Rows* and in the *Columns*. You can create a maximum of two structures per query. Only one of these structures can contain key figures.

Context

There are various ways of creating a structure:

- Choose a key figure from the *BW Repository* and drag it to either the *Rows* area or the *Columns* area. The system automatically generates a new structure containing the selected key figure. This structure has the description *Key Figures*.
- Choose *New Structure* in the context menu for either *Rows* or *Columns*.

Related Information

Defining Structure Elements [page 328]
Defining Priorities [page 360]

4.10.2.2.1 Structure Properties

If you have inserted a structure into the rows or columns of a query when defining the query, the system displays the structure's properties in the right-hand screen area.

- Under *General*, you can set the technical name and enter a description.
- Under *Display*, you can make settings regarding *Visibility of Structure Member* and *Zero Suppression*.

4.10.2.3 Check Structure

Since structures in the query definition are often complex, a structure overview is available. You can open the window with the structure overview from the *Overview* context menu on the force element.

Context

For a complete overview of a structure, you can call *Expand* from the context menu.

The *Default Text Setting* column contains the setting for creating standard texts. For restricted key figures for example, the system uses the first restriction as the value for text creation by default.

The system determines this setting automatically during query definition and uses it at runtime to generate the text of the structure element in case no text has been maintained manually.

Procedure

1. Check the values by choosing *Check And Correct* from the context menu.

   If values do not match the default values, you can prompt the system to make a suitable proposal for the model. You can accept the proposal by choosing *Change* or can reject it by choosing *Cancel*. The system highlights adjusted values in color in the window.
2. You can change the technical names by double-clicking them.
3. You can add a comment to a query element. This text is displayed as a quick fix in the *Query Monitor* transaction (RSRT).

4.10.2.4 Defining Structure Elements

Objects, which can only be defined in structures, are selections and formulas.

Context

A structure is required in order to define a structure element. Structure elements can be hierarchically arranged in a structure. They have common properties.

Related Information

Structure Element [page 113]
Defining Selections [page 329]
Defining a Formula [page 344]

4.10.2.4.1 Defining Selections

You can define selections with and without key figures.

Context

You are on the *Sheet Definition* tab in the query definition.

Procedure

1. Define a **selection with key figures** if this is required for your query definition.
   a. Choose a key figure from the *BW Repository* and drag it to either the *Rows* area or the *Columns* area. The system automatically generates a new structure containing the selected key figure. This structure has the description *Key Figures*. The system automatically sets the selection for this key figure.

      Note that you can have precisely **one** key figure structure per query. If you want to use more than one key figure, the key figure structure must contain all of these.
   b. To restrict a key figure, open the context menu for the key figure in question and choose *Edit*. The *Edit Selection* dialog box appears. In the properties of the key figure, you see the selection in the *General* tab under *Selection Details*.

      Note that you can only use a key figure restricted in this way in the current query. You can find all restricted key figures used in queries for an InfoProvider by choosing *Reusable Components* > *Restricted Key Figures* in the *BW Repository*.
2. Define a **selection without key figures** if this is required for your query definition.
   a. Choose *New Selection* from the context menu for the required structure. The *Edit Selection* dialog box appears.
   b. Add the required characteristic.
   c. Select the required characteristic values and confirm your selection by choosing *OK*.

Related Information

Selection [page 114]
Selection Properties [page 330]
Restricting Key Figures [page 342]
Restrict Formula [page 343]
Defining Restricted Key Figures [page 382]
This link takes to you to the documentation for reusable query components.

4.10.2.4.1.1 Selection Properties

If you have inserted a key figure as part of a structure in the rows and columns of a query during definition, the system creates a selection. You can edit the properties of the selection on the right of the screen. If you have defined a selection without key figures, you can edit the properties of this selection in the right of the screen.

The *Properties* screen area is divided into tab pages, each of which provides different settings.

> ⚠ Caution
>
> Note that you can activate manual input for the settings by using ✎.
>
> If you want to use a default value specified on the object again, choose ✎. The system displays the default settings as non-input-ready fields.

Related Information

Tab: General [page 330]
Tab: Conversion [page 333]
Tab: Calculations [page 333]

4.10.2.4.1.1.1 Tab: General

On the *General* tab, you can view the properties of the selection and make various settings.

General

This area displays the technical name and the description.

You can change the description. The new description text is different to the text in InfoObject maintenance and is used in the query display.

> ⓘ Note
>
> In the description, you can use text variables, which the user can enter at query runtime.
>
> - If you want to use one of the existing text variables in the description, choose *Add Text Variable*. The dialog box for choosing text variables appears. Mark the selected text variable and confirm with *OK*. The system inserts the selected text variable into the description. The text variable starts and ends with the character &.
> - If you want to create a new text variable, start with the character &. Using the key combination CTRL + space you can display the input help with all the variable names available in the system. You can also enter a new name. End the name with the character &.

Constant Selection

The **Constant Selection** property is available for selections in structural components, cells and restricted key figures in queries.

If you go to the *General* tab and then go to the *Constant Selection* screen area and select **total** as constant, this means that this selection **cannot** be used at runtime for navigation and filtering. Therefore this selection can serve as a reference size.

You can also define components of selections - individual characteristics and their filter values - as constant. To do this, open the *Edit Selection* dialog box, go to the *Edit Selection* screen and choose *Constant Selection* from the context menu of the selection component. During navigation, the selection only remains unaffected by filters with relation to this characteristic.

If the specifications for constant selection overlap (in other words, if **entire selections** have been defined as constant on structure components and on cells, and if components of these same selections have been **individually** defined as constant), then the following rule applies: Constant selection on a selection element takes precedence over the constant selection on the characteristic defined on selections in this structure element.

Display

Under *Display As*, you can specify whether the key figure values of the selection are displayed, and how. You can choose here from the options created for the key figure in InfoObject maintenance. The following table provides an overview of the possible display options in the BW system:

Display Type

Display Type	Description
Hide	Hide: If you choose *Always Show*, the key figure is always visible. If you choose *Always Hide* , the key figure is always hidden in the executed report. If you choose *Hide (Can be Shown)*, the key figure is initially hidden in the executed report but can be displayed again from within the report. You can use this setting to ensure that only information that is really necessary is displayed in the start view for the report. You can also show additional details if required.
Highlight	Highlight: You can choose whether to show the key figure using *Normal Display* or *Highlighted Display*.

Display Type	Description
Number of Decimal Places	Number of decimal places: You can specify the number of decimal places. Numbers can have any number of decimal places from zero up to nine decimal places (0, 0.0, 0.00... to 0.000000000).
Scaling Factor	Scaling factor: You can specify a scaling factor between one and one billion. If you set 1000 for example, the value 3000 is displayed in the report as 3.
Change Sign	Change sign: Specify whether you want to keep the signs (*Keep +/- Sign*) or reverse the signs (*Reverse Sign*). If you choose *Reverse Sign*, a positive number is given a minus sign (85 becomes -85, for example) and a negative number is given a plus sign (-38 becomes +38, for example). Note that reserve sign is purely a display function. If the key figure is added to a formula, for example, it is calculated with its correct values.
Node Status	Node status: If you have arranged the structure elements in hierarchical way, you can specify how the node is displayed in the structure: (*expanded*) or (*collapsed*). This property applies to the executed query and is unaffected by the status of the node in query definition.

Selection Details

This area displays the selection. To change the selection, choose *Edit*. The *Edit Selection* dialog box appears.

Related Information

Restricting Key Figures [page 342]

4.10.2.4.1.1.2 Tab: Conversion

On the *Conversion* tab, you can configure various settings for converting currencies and units.

Currency Translation

Here you can select a currency translation type and a target currency for a structural component.

You can choose from the currency translation types that you created in your BW system. Choose *SAP Menu* > *Modeling* > *Object Maintenance* > *Currency Translation Types*.

Unit Conversion

Here you can select a unit conversion type and a target unit for a structural component.

You can choose from the unit conversion types that you created in your BW system. Choose *SAP Menu* > *Modeling* > *Object Maintenance* > *Unit Conversion Types*.

4.10.2.4.1.1.3 Tab: Calculations

On the *Calculations* tab, you can configure local calculation settings for a formula or a key figure.

Local calculations are performed (on the client) on the result set provided by your BW system. This allows you to recalculate results or single values from a report based on specific criteria. On the *Calculations* screen, you can specify for a structure component which rules are used to recalculate result rows and/or single values, and in which direction the calculations are performed.

> ⓘ Note
>
> For example, you can create ranking lists or calculate the total of a top 10 product list.

Note the following features:

- Local calculations only work on the displayed data. This means that local calculations only change how the values are displayed. For any subsequent calculations (such as formulas), the system uses the original values calculated by the BW system and **NOT** the display values changed by the local calculations.
- Local calculations **cannot** be applied to hierarchies.
- If you are processing large result sets, local calculations can take significantly more time to perform.

  > ⓘ Note
  >
  > For performance reasons, we recommend that you work with exception aggregation and calculated key figures. This is because all the operations executed on the server can be processed significantly faster (especially when using the SAP HANA database).

Related Information

Calculate Results As [page 334]
Example: Calculate Results As [page 335]
Calculate Single Values As [page 336]
Examples: Calculate Single Values As [page 339]

4.10.2.4.1.1.3.1 Calculate Results As

Calculate Result As allows you to select which rule to use to locally recalculate the results of a report.

Note that the functions listed below are not fully supported by all clients.

You can choose from the following settings:

Settings	Description
(Not defined)	This setting displays the results provided by the analytic manager. No local calculation takes place.
Average	The average (total divided by total number of values) of all values for the corresponding area is displayed as the result.
Average of Detailed Values That are Not Zero, Null or Error	The average (total divided by number) of all values for the corresponding area is displayed as the result, with the exception of zeroes, meaning that values equal to zero are **not** counted.
Counter of All Detailed Values	The number of values for the corresponding area is displayed as the result.
Counter of All Detailed Values That are Not Zero, Null or Error	The number of values for the corresponding area, excluding zeroes, is displayed as the result; values equal to zero are **not** included.
First Value	The highest value or the value furthest to the left of the corresponding area is displayed as the result.
Hide	The result is **not** displayed.
Last Value	The lowest value or the value furthest to the right of the corresponding area is displayed as the result.
Maximum	The largest value for the corresponding area is displayed as the result.
Minimum	The smallest value for the corresponding area is displayed as the result.

Settings	Description
Standard Deviation	Statistical deviation is a measure of the distribution of the values with respect to the mean value (average). Statistical deviation = root from the variance: SQRT (variance).
Summation	The sum total of the values for the corresponding area is displayed as the result.
Summation of Rounded Values	If you have set a scaling factor, it is useful to calculate the total sum of rounded values locally since the total sums can differ considerably (in particular with high scaling factors). **Note** If the result is a totals row, the area calculated is the corresponding column. If the result is a totals column, the area calculated is the corresponding row. With higher totals, multiple drilldowns (with results and an overall result) that is, all single values affected are taken into account (without results).
Variance	The following formula is used to calculate the statistical variance: (å x² - (å x)² / N) / (N-1) for N > 1 and notexist for N=1

Related Information

Example: Calculate Results As [page 335]

4.10.2.4.1.1.3.1.1 Example: Calculate Results As

The following example serves to illustrate the significance of the *Summation of Rounded Values* function. The first table shows the values of a simple query in which sales have been set with a scaling factor of 1,000:

Customer	Sales
	EUR 1,000
Customer 1	2.246
Customer 2	2.126

Customer	Sales
	EUR 1,000
Total	**4.372**

If you reduce the number of decimal places, the values are rounded:

Customer	Sales
	EUR 1,000
Customer 1	2.25
Customer 2	2.13
Total	**4.37**

You can see that the total displayed locally is not the value that you would expect. This value was calculated by first making the calculation and the displaying and rounding the values.

If you use the *Summation of Rounded Values* function, the values are first rounded and then calculated and displayed. The query then appears as follows:

Customer	Sales
	EUR 1,000
Customer 1	2.25
Customer 2	2.13
Total	**4.38**

Related Information

Calculate Results As [page 334]
Example: Calculate Results As [page 335]

4.10.2.4.1.1.3.2 Calculate Single Values As

Calculate Result As allows you to select which rule to use to locally recalculate individual values of a report.

Note that the functions listed below are not fully supported by all clients.

You can choose from the following settings:

Settings	Description
(Not defined)	This setting displays the data provided by the analytic manager. No local calculation of single values takes place.
Hide	The system only displays the results; the single values are suppressed.
Maximum	The system displays the maximum key figure value for a characteristic.
Minimum	The system displays the minimum key figure value for a characteristic.
Moving Average	The system calculates the average of all values.
Moving Average that is Not Zero, Null or Error	The system calculates the average of all values, but ignores any zero values.
Normalize According to Next Group Level Result	The data is displayed as a percentage of the result of the next group level. The values of the results row and the overall results row are not displayed as percentages but as absolute values. If there is only one characteristic in the drilldown, the result of a group level is the same as the overall result.
Normalize According to Overall Result	The data is displayed as a percentage of the overall result. The values of the results row and the overall results row are not displayed as percentages but as absolute values. If there are multiple characteristics in the drilldown, there are different results, which are combined to form an overall result.
Normalize According to Unrestricted Overall Result	The data is displayed as a percentage of the unrestricted result (query result). The values of the results row and the overall results row are not displayed as percentages but as absolute values. Each key figure for a query has a query result. This is the result of the key figure that results from the aggregation of all characteristics for the query. The overall result and the query result are identical to one another, provided that none of the characteristics are filtered at runtime.

Settings	Description
Olympic Rank Number	The Olympic ranked list differs from the basic ranked list as follows: In the Olympic ranked list, when a value occurs more than once, the next smallest value is not assigned the rank incremented by one, but the rank that corresponds to the number of previous characteristic values (including the current value). → Tip There are three products with a higher rank than product D, therefore D has rank 4 and rank 3 is not assigned since B and C have the same rank (2). A 100 = Rank 1 B 50 = Rank 2 C 50 = Rank 2 D 20 = Rank 4
Rank Number	The characteristic values are sorted according to the selected structure element and are given a ranking. The order of the ranked list is based on the size of the value for the structure element, where the largest value has rank 1 and the smallest value has the last rank. If a value occurs more than once, the corresponding characteristic values are assigned the same rank. In a basic ranked list, the next smallest value is assigned this rank incremented by one. → Tip You can create a ranked list of products according to sales. A 100 = Rank 1 B 50 = Rank 2 C 50 = Rank 2 D 20 = Rank 3

Related Information

Examples: Calculate Single Values As [page 339]

4.10.2.4.1.1.3.2.1 Examples: Calculate Single Values As

Overview of Functions

The following table provides an overview of the following functions: *Normalize According to Next Group Level Result*, *Normalize According to Overall Result* and *Normalize According to Unrestricted Overall Result*:

Customer	Sales	*Normalize According to Next Group Level Result*	*Normalize According to Overall Result*	*Normalize According to Unrestricted Overall Result*
Customer 1	10	10.86 %	0.97 %	0.12 %
Customer 2	0	00.00 %	0.00 %	0.00 %
Customer 3	17	18.47 %	1.66 %	0.21 %
Customer 4	8	08.69 %	0.78 %	0.10 %
Customer 5	15	16.30 %	1.46 %	0.19 %
Customer 6	22	23.91 %	2.15 %	0.28 %
Customer 7	15	16.30 %	1.46 %	0.19 %
Result	**92**			
Overall Result	**1023**			
Query Result	**7812**			

The following table provides an overview of the functions *Rank Number* and *Olympic Rank Number*:

Customer	Sales	*Rank Number*	*Olympic Rank Number*
Customer 1	10	4	5
Customer 2	0	6	7
Customer 3	17	2	2
Customer 4	8	5	6
Customer 5	15	3	3
Customer 6	22	1	1
Customer 7	15	3	3

The following table provides an overview of the functions *Maximum* and *Minimum*:

Customer	Sales	*Maximum*	*Minimum*
Customer 1	10	22	0
Customer 2	0	22	0
Customer 3	17	22	0
Customer 4	8	22	0
Customer 5	15	22	0
Customer 6	22	22	0
Customer 7	15	22	0

The following table provides an overview of the functions *Moving Average*, *Moving Average that is Not Zero, Null or Error* and *Hide*:

Customer	Sales	*Moving Average*	*Moving Average that is Not Zero, Null or Error*	*Hide*
Customer 1	10	10.00	10.00	
Customer 2	0	5.00	10.00	
Customer 3	17	9.00	13.50	
Customer 4	8	8.75	11.66	
Customer 5	15	10.00	12.50	
Customer 6	22	12.00	14.40	
Customer 7	15	12.43	14.50	

Calculation direction

The **calculation direction** is always from the top in the direction of the drilldown.

- In this case (characteristics in the rows and key figures in the column), the calculation direction is from top to bottom.
- If the key figures are in the rows and the characteristics are in the columns, the calculation direction is from left to right.
- If a query contains characteristics in the rows and in the columns and the query is filtered by a key figure (for example, sales), the calculation begins at the top left and works down to the bottom right.

Example query with Sales in the filter and Customer and Region in the drilldown:

Customer	Germany	England	France	Result
BMW	10	17	35	62
VW	0	15	17	32
Mercedes	12	22	3	37
Result	22	54	55	131

The following table shows the query with the function *Normalize According to Overall Result*:

Customer	Germany	England	France
BMW	7.63 %	12.97 %	26.71 %
VW	0.00 %	11.45 %	12.97 %
Mercedes	9.16 %	16.79 %	2.29 %

The following table shows the query with the *Rank Number* function.

Customer	Germany	England	France
BMW	6	3	1
VW	8	4	3
Mercedes	5	2	7

The following table shows the query with the *Olympic Rank Number* function:

Customer	Germany	England	France
BMW	7	3	1
VW	9	5	3
Mercedes	6	2	8

Related Information

Calculate Single Values As [page 336]

4.10.2.4.1.2 Restricting Key Figures

You can restrict a key figure using one or more characteristic selections. This key figure can a basic key figure, a calculated key figure or a key figure figure that is already restricted. The key figure that you restrict can only be used in the context of the current query.

Context

You are in the *Edit Selection* screen for the selected key figure.

Procedure

1. Here you can enter a new technical name and description for your restricted key figure. If you do not specify anything new, the key figure retains its existing technical name and description.
2. The left side of the lower area displays the directory of all the available objects for the InfoProvider. The *Selection Details* area on the right side displays the selected key figure. Select the InfoObject that you want to restrict and then drag and drop it onto the *Selection Details* area.

Example

You have selected the "Sales" key figure and you want to restrict it to the year 2013.

1. To do this, choose "Calendar Year" (`0CALYEAR`) from the time characteristics in the InfoProvider directory. Drag and drop this characteristic onto the "Sales" key figure.
2. Open the context menu of the "Calendar Year" characteristic and choose *Restrict*. The *Edit Filter* dialog box appears.
3. Select the required characteristic value - in our example "2013" - and choose *Add*.
4. Confirm your filter definition by pressing *OK*.
5. Confirm your restriction definition by pressing *OK*.

Related Information

Defining Filters [page 362]

4.10.2.4.1.3 Restrict Formula

In a key figure structure, you can restrict a local formula in a further selection. The restricted selection can only be used in the context of the current query.

Context

You are in your query's *Sheet Definition*.

Procedure

1. First create a new selection.
2. Now drag the formula that you want to restrict to this selection.
3. To restrict the formula, open the context menu from the key figure and choose *Edit*. The *Edit Selection* dialog appears. Here you can enter a new technical name and description for your restricted formula. If you do not make any entries here, the key figure retains its existing technical name and description.
4. The left side of the lower area displays the directory of all the available objects for the InfoProvider. The system displays the available local formulas under the *Formulas* node. Under *Selection Details* on the right side of the screen, you can see the selected key figure or formula.
5. Select the formula that you want to restrict and then drag and drop it onto the *Selection Details* area.
6. Select the characteristic value that you want to restrict and then drag and drop it onto the *Selection Details* area.
7. Choose *OK*.

Results

The system inserts the new formula, that is key figure-based and restricted by the characteristic value. into the key figure structure.

Example

You have selected the *Average Price* formula and want to restrict it to country *Germany*.

1. To do this, choose country `COUNTRY` from the characteristic of the InfoProvider directory, choose *Germany* here in our example, and drag and drop to drag it this the formula.
2. Confirm the definition of your restriction by pressing *OK*.

4.10.2.4.2 Defining a Formula

You can recalculate the key figures in a structure using a formula. This allows you to obtain calculated key figures. You can include basic key figures, restricted key figures, calculated key figures, and reusable key figures in the formula definition.

Context

You are on the *Sheet Definition* tab in the query definition.

Procedure

1. If the query does not have a key figure structure yet, create a New Structure in the Rows or Columns by selecting the row or column directory using the right mouse button and choosing *New Structure* from the context menu.
2. If the structure does not contain any key figures yet, drag a (basic) key figure from the InfoProvider to the directory of the new structure.
3. Select the structure directory and choose *New Formula* from the context menu. The *Create Formula* dialog box appears.
4. You can enter a technical name and description for your formula here. If you do not do this, the formula is given the description *Formula n*, with "n" corresponding to the number of formulas per query.

   In the description, you can use text variables, which the user can enter at query runtime.

   - If you want to use one of the existing text variables in the description, choose *Add Text Variable*. The dialog box for choosing text variables appears. Mark the selected text variable and confirm with *OK*. The system inserts the selected text variable into the description. The text variable starts and ends with the character &.
   - If you want to create a new text variable, start with the character &. Using the key combination CTRL + space you can display the input help with all the variable names available in the system. You can also enter a new name. End the name with the character &.
5. Define the formula. You can use input help for this. To simplify the task of formula definition, you can select the required operands and operators for the formula from *Groups*. The system then inserts these into the formula definition with the correct syntax.
   a. From *Groups*, select the operands required for the formula definition. You can choose from all structure elements that you have assigned to the structure so far.

      Choose *Groups* > *Structure: Key Figures*. Das System zeigt im Bildbereich *Formula Elements* die Kennzahlen Sales und Quantity. Sie markieren Sales und wählen *Add*.
   b. Choose the required operation for the formula definition. If it is a basic arithmetic operation, you can enter it using the numeric keypad on the right. You can select other operators from the *Groups*. You can choose from the following function types here:
      - *Percentage Functions*
      - *Data Functions*

- *Mathematical Functions*
- *Trigonometrical Functions*
- *Boolean Operators*

Select the symbol for dvision as operator on the numeric keypad on the right. In our example, we still need the second key figure Quantity as the divisor. We insert this as described above.
With each step, the system checks the syntactic correctness of the formula. If the formula is incomplete or syntactically incorrect, the system underlines the error in red.

c. Specify any numerical values you require for the formula by entering them on the numeric keypad.

> **Note**
>
> You can also enter digits, parentheses and basic arithmetical operations using the keyboard.

d. Choose *Format Formula* to format your formula. In this step for example, the system removes any unneeded spaces and inserts any line breaks required.

You can also change the syntax highlighting for the formula. To do this, choose *Window* > *Preferences* > *BW Modeling* > *Query Formula Syntax Coloring*, and configure the required colors, styles and fonts.

6. Confirm your formula definition by pressing *OK*. You are back on the *Sheet Definition* tab in the query definition.
7. Make any necessary settings for the properties of the formula on the right of the screen.

Related Information

Formula [page 114]
Formula Calculation [page 583]
Basic Functions [page 584]
Percentage Functions [page 584]
Data Functions [page 587]
Mathematical Functions [page 598]
Trigonometric functions [page 600]
Boolean Operators [page 601]
Formula Properties [page 346]
Tab: General [page 346]
Tab Page: Aggregation [page 349]
Tab: Calculations [page 333]
Defining Calculated Key Figures [page 379]

4.10.2.4.2.1 Formula Properties

If you have inserted a formula - as part of a structure - into the rows or columns of a query when defining the query, the system displays the formula properties in the right-hand screen area.

The *Properties* screen area is divided into tab pages, each of which provides different settings.

> ⚠ Caution
>
> Note that you can activate manual input for the settings by using .
>
> If you want to use a default value specified on the object again, choose . The system displays the default settings as non-input-ready fields. The default values for a formula do not depend on the settings of the InfoObjects involved in the formula.

Related Information

Tab: General [page 346]
Tab Page: Aggregation [page 349]
Tab: Calculations [page 333]

4.10.2.4.2.1.1 Tab: General

On the *General* tab, you can view the formula properties and make various settings.

General

This area displays the technical name and the description.

You can change the description. The new description text is different to the text specified for the key figure in InfoObject maintenance and is used for presentation in the query.

Formula Collision

Whenever you define two structures that both contain formulas, it may be unclear to the system how to calculate the formulas at the point where both formulas intersect. Collisions always occur when multiplication/division and addition/subtraction are mixed in competing formulas. If both formulas involve multiplication/division or addition/subtraction only, both calculation directions return the same result.

Example

Example of Formula Collision

	Column 1	Column 2	Column 1 x Column 2
Row 1	Value A	Value B	AxB
Row 2	Value C	Value D	CxD
Row 1 + Row 2	A+C	B+D	Formula Collision

In the cell in which the row formula and column formula meet, there is a formula collision: If you calculate according to the column formula in this cell, the cell contains (A+C)x(B+D). If you calculate according to the row formula in this cell, the cell contains (AxB)+(CxD). The result achieved therefore depends on the type of calculation performed.

When a formula collision occurs, you can specify how the formula is calculated.

Resolving Formula Collisions

Option	Description
(Default value)	If you do not specify a value, the formula that was last specified and saved takes priority in a formula collision.
Use result of this formula	Use result of this formula: The result of this formula has priority in a collision.
Use result of concurrent formula	Use result of concurrent formula The result of a competing formula has priority in a collision.

Display

Under *Display As*, you can specify whether the formula is displayed and how. The following table provides an overview of the possible display options in the BW system:

Display Type

Display Type	Description
Hide	Hide: If you choose *Always Show*, the key figure is always visible. If you choose *Always Hide* , the key figure is always hidden in the executed report. If you choose *Hide (Can be Shown)*, the key figure is initially hidden in the executed report but can be displayed again from within the report. You can use this setting to ensure that only information that is really necessary is displayed in the start view for the report. You can also show additional details if required.

Display Type	Description
Highlight	Highlight: You can choose whether to show the key figure using *Normal Display* or *Highlighted Display*.
Number of Decimal Places	Number of decimal places: You can specify the number of decimal places. Numbers can have any number of decimal places from zero up to nine decimal places (0, 0.0, 0.00... to 0.000000000).
Scaling Factor	Scaling factor: You can specify a scaling factor between one and one billion. If you set 1000 for example, the value 3000 is displayed in the report as 3.
Change Sign	Change sign: Specify whether you want to keep the signs (*Keep +/- Sign*) or reverse the signs (*Reverse Sign*). If you choose *Reverse Sign*, a positive number is given a minus sign (85 becomes -85, for example) and a negative number is given a plus sign (-38 becomes +38, for example). Note that reserve sign is purely a display function. If the key figure is added to a formula, for example, it is calculated with its correct values.
Node Status	Node status: If you have arranged the structure elements in hierarchical way, you can specify how the node is displayed in the structure: (*expanded*) or (*collapsed*). This property applies to the executed query and is unaffected by the status of the node in query definition.

Formula

This area displays the formula.

To change the formula definition, choose *Edit*. The *Edit Formula* dialog box appears.

4.10.2.4.2.1.2 Tab Page: Aggregation

On the *Aggregation* tab, you can create an exception aggregation for a formula and select or exclude reference characteristics.

Related Information

Rules for Standard Aggregation and Exception Aggregation [page 579]

4.10.2.4.2.1.3 Tab: Calculations

On the *Calculations* tab, you can configure local calculation settings for a formula or a key figure.

Local calculations are performed (on the client) on the result set provided by your BW system. This allows you to recalculate results or single values from a report based on specific criteria. On the *Calculations* screen, you can specify for a structure component which rules are used to recalculate result rows and/or single values, and in which direction the calculations are performed.

> ⓘ Note
>
> For example, you can create ranking lists or calculate the total of a top 10 product list.

Note the following features:

- Local calculations only work on the displayed data. This means that local calculations only change how the values are displayed. For any subsequent calculations (such as formulas), the system uses the original values calculated by the BW system and **NOT** the display values changed by the local calculations.
- Local calculations **cannot** be applied to hierarchies.
- If you are processing large result sets, local calculations can take significantly more time to perform.

  > ⓘ Note
  >
  > For performance reasons, we recommend that you work with exception aggregation and calculated key figures. This is because all the operations executed on the server can be processed significantly faster (especially when using the SAP HANA database).

Related Information

Calculate Results As [page 334]
Example: Calculate Results As [page 335]
Calculate Single Values As [page 336]
Examples: Calculate Single Values As [page 339]

4.10.2.4.2.1.3.1 Calculate Results As

Calculate Result As allows you to select which rule to use to locally recalculate the results of a report.

Note that the functions listed below are not fully supported by all clients.

You can choose from the following settings:

Settings	Description
(Not defined)	This setting displays the results provided by the analytic manager. No local calculation takes place.
Average	The average (total divided by total number of values) of all values for the corresponding area is displayed as the result.
Average of Detailed Values That are Not Zero, Null or Error	The average (total divided by number) of all values for the corresponding area is displayed as the result, with the exception of zeroes, meaning that values equal to zero are **not** counted.
Counter of All Detailed Values	The number of values for the corresponding area is displayed as the result.
Counter of All Detailed Values That are Not Zero, Null or Error	The number of values for the corresponding area, excluding zeroes, is displayed as the result; values equal to zero are **not** included.
First Value	The highest value or the value furthest to the left of the corresponding area is displayed as the result.
Hide	The result is **not** displayed.
Last Value	The lowest value or the value furthest to the right of the corresponding area is displayed as the result.
Maximum	The largest value for the corresponding area is displayed as the result.
Minimum	The smallest value for the corresponding area is displayed as the result.
Standard Deviation	Statistical deviation is a measure of the distribution of the values with respect to the mean value (average). Statistical deviation = root from the variance: SQRT (variance).
Summation	The sum total of the values for the corresponding area is displayed as the result.

Settings	Description
Summation of Rounded Values	If you have set a scaling factor, it is useful to calculate the total sum of rounded values locally since the total sums can differ considerably (in particular with high scaling factors). **Note** If the result is a totals row, the area calculated is the corresponding column. If the result is a totals column, the area calculated is the corresponding row. With higher totals, multiple drilldowns (with results and an overall result) that is, all single values affected are taken into account (without results).
Variance	The following formula is used to calculate the statistical variance: (å x² - (å x)² / N) / (N-1) for N > 1 and notexist for N=1

Related Information

Example: Calculate Results As [page 335]

4.10.2.4.2.1.3.1.1 Example: Calculate Results As

The following example serves to illustrate the significance of the *Summation of Rounded Values* function. The first table shows the values of a simple query in which sales have been set with a scaling factor of 1,000:

Customer	Sales
	EUR 1,000
Customer 1	2.246
Customer 2	2.126
Total	**4.372**

If you reduce the number of decimal places, the values are rounded:

Customer	Sales
	EUR 1,000
Customer 1	2.25
Customer 2	2.13
Total	**4.37**

You can see that the total displayed locally is not the value that you would expect. This value was calculated by first making the calculation and the displaying and rounding the values.

If you use the *Summation of Rounded Values* function, the values are first rounded and then calculated and displayed. The query then appears as follows:

Customer	Sales
	EUR 1,000
Customer 1	2.25
Customer 2	2.13
Total	**4.38**

Related Information

Calculate Results As [page 334]
Example: Calculate Results As [page 335]

4.10.2.4.2.1.3.2 Calculate Single Values As

Calculate Result As allows you to select which rule to use to locally recalculate individual values of a report.

Note that the functions listed below are not fully supported by all clients.

You can choose from the following settings:

Settings	Description
(Not defined)	This setting displays the data provided by the analytic manager. No local calculation of single values takes place.
Hide	The system only displays the results; the single values are suppressed.

Settings	Description
Maximum	The system displays the maximum key figure value for a characteristic.
Minimum	The system displays the minimum key figure value for a characteristic.
Moving Average	The system calculates the average of all values.
Moving Average that is Not Zero, Null or Error	The system calculates the average of all values, but ignores any zero values.
Normalize According to Next Group Level Result	The data is displayed as a percentage of the result of the next group level. The values of the results row and the overall results row are not displayed as percentages but as absolute values. If there is only one characteristic in the drilldown, the result of a group level is the same as the overall result.
Normalize According to Overall Result	The data is displayed as a percentage of the overall result. The values of the results row and the overall results row are not displayed as percentages but as absolute values. If there are multiple characteristics in the drilldown, there are different results, which are combined to form an overall result.
Normalize According to Unrestricted Overall Result	The data is displayed as a percentage of the unrestricted result (query result). The values of the results row and the overall results row are not displayed as percentages but as absolute values. Each key figure for a query has a query result. This is the result of the key figure that results from the aggregation of all characteristics for the query. The overall result and the query result are identical to one another, provided that none of the characteristics are filtered at runtime.

Settings	Description
Olympic Rank Number	The Olympic ranked list differs from the basic ranked list as follows: In the Olympic ranked list, when a value occurs more than once, the next smallest value is not assigned the rank incremented by one, but the rank that corresponds to the number of previous characteristic values (including the current value). → Tip There are three products with a higher rank than product D, therefore D has rank 4 and rank 3 is not assigned since B and C have the same rank (2). A 100 = Rank 1 B 50 = Rank 2 C 50 = Rank 2 D 20 = Rank 4
Rank Number	The characteristic values are sorted according to the selected structure element and are given a ranking. The order of the ranked list is based on the size of the value for the structure element, where the largest value has rank 1 and the smallest value has the last rank. If a value occurs more than once, the corresponding characteristic values are assigned the same rank. In a basic ranked list, the next smallest value is assigned this rank incremented by one. → Tip You can create a ranked list of products according to sales. A 100 = Rank 1 B 50 = Rank 2 C 50 = Rank 2 D 20 = Rank 3

Related Information

Examples: Calculate Single Values As [page 339]

4.10.2.4.2.1.3.2.1 Examples: Calculate Single Values As

Overview of Functions

The following table provides an overview of the following functions: *Normalize According to Next Group Level Result*, *Normalize According to Overall Result* and *Normalize According to Unrestricted Overall Result*:

Customer	Sales	*Normalize According to Next Group Level Result*	*Normalize According to Overall Result*	*Normalize According to Unrestricted Overall Result*
Customer 1	10	10.86 %	0.97 %	0.12 %
Customer 2	0	00.00 %	0.00 %	0.00 %
Customer 3	17	18.47 %	1.66 %	0.21 %
Customer 4	8	08.69 %	0.78 %	0.10 %
Customer 5	15	16.30 %	1.46 %	0.19 %
Customer 6	22	23.91 %	2.15 %	0.28 %
Customer 7	15	16.30 %	1.46 %	0.19 %
Result	**92**			
Overall Result	**1023**			
Query Result	**7812**			

The following table provides an overview of the functions *Rank Number* and *Olympic Rank Number*:

Customer	Sales	*Rank Number*	*Olympic Rank Number*
Customer 1	10	4	5
Customer 2	0	6	7
Customer 3	17	2	2
Customer 4	8	5	6
Customer 5	15	3	3
Customer 6	22	1	1
Customer 7	15	3	3

The following table provides an overview of the functions *Maximum* and *Minimum*:

Customer	Sales	*Maximum*	*Minimum*
Customer 1	10	22	0
Customer 2	0	22	0
Customer 3	17	22	0
Customer 4	8	22	0
Customer 5	15	22	0
Customer 6	22	22	0
Customer 7	15	22	0

The following table provides an overview of the functions *Moving Average*, *Moving Average that is Not Zero, Null or Error* and *Hide*:

Customer	Sales	*Moving Average*	*Moving Average that is Not Zero, Null or Error*	*Hide*
Customer 1	10	10.00	10.00	
Customer 2	0	5.00	10.00	
Customer 3	17	9.00	13.50	
Customer 4	8	8.75	11.66	
Customer 5	15	10.00	12.50	
Customer 6	22	12.00	14.40	
Customer 7	15	12.43	14.50	

Calculation direction

The **calculation direction** is always from the top in the direction of the drilldown.

- In this case (characteristics in the rows and key figures in the column), the calculation direction is from top to bottom.
- If the key figures are in the rows and the characteristics are in the columns, the calculation direction is from left to right.
- If a query contains characteristics in the rows and in the columns and the query is filtered by a key figure (for example, sales), the calculation begins at the top left and works down to the bottom right.

Example query with Sales in the filter and Customer and Region in the drilldown:

Customer	Germany	England	France	Result
BMW	10	17	35	62
VW	0	15	17	32
Mercedes	12	22	3	37
Result	22	54	55	131

The following table shows the query with the function *Normalize According to Overall Result*:

Customer	Germany	England	France
BMW	7.63 %	12.97 %	26.71 %
VW	0.00 %	11.45 %	12.97 %
Mercedes	9.16 %	16.79 %	2.29 %

The following table shows the query with the *Rank Number* function.

Customer	Germany	England	France
BMW	6	3	1
VW	8	4	3
Mercedes	5	2	7

The following table shows the query with the *Olympic Rank Number* function:

Customer	Germany	England	France
BMW	7	3	1
VW	9	5	3
Mercedes	6	2	8

Related Information

Calculate Single Values As [page 336]

4.10.2.5 Defining Exception Cells and Help Cells

A cell is the cross product of two structural elements. On the *Cells* tab, you can specify whether a cell is used as an exception cell or help cell.

Prerequisites

The query contains two structures.

Context

The system automatically adds the *Cells* tab.

- In the *Cells* screen area, you can define exception cells.
- In the *Help Cells* screen area, you can define help cells.
- The *Properties* screen area displays the properties of the cell, which match the properties of the relevant structure element.

Procedure

1. To define an exception cell, select the required cell in the *Cells* area.
   - To set the cell as a reference cell, choose *New Reference Cell* from the context menu.
   - To set the cell as a formula, choose *New Formula Cell* from the context menu.
   - To set the cell as a selection, choose *New Selection Cell* from the context menu.
2. To define a help cell, choose the required option from the context menu in the *Help Cells* screen area:
   - To create a help formula cell, choose *New Formula Cell* from the context menu.
   - To create a help selection cell, choose *New Selection Cell* from the context menu.

Related Information

Cell, Exception Cell and Help Cell [page 114]
Defining Cell References [page 359]
Defining Help Cells [page 359]
Defining Selections [page 329]
Defining a Formula [page 344]
Defining Priorities [page 360]

4.10.2.5.1 Defining Cell References

You can use a reference cell to reuse the implicit cell definition in a formula, without needing to manually create this value by using a new selection.

Context

You have a query with two structures and you want to reuse the value of the implicitly generic cell definition, which is created at the intersection of two structure elements.

Procedure

1. Select the required cell in the *Cells* screen area.
2. Open the context menu and choose *New Reference Cell*. A description based on the two structural components now automatically appears in the cell. The symbol shows you that this exception cell is a reference cell.
3. Specify the properties of the reference cell in the *Properties* screen area.
    - On the *General* tab, you can change the technical name, description and display properties of a defined exception cell.
    - On the *Calculations* tab, you can specify local calculations.

4.10.2.5.2 Defining Help Cells

The following example shows how an additional cell created in the *Help Cells* screen area can be used as a help selection for an exception cell of type "formula". The help cell itself is **not** displayed in the query.

Context

You can define an invoice in a help cell of type "selection" and use this in an exception cell of type "formula", in order to calculate a duplicated invoice.

Procedure

1. Open the context menu for the empty cell in the *Help Cells* area and choose *New Selection Cell*. The *Edit Selection* dialog box appears.

2. Enter a technical name for the help cell. You can also enter a new description.
3. Use drag and drop to move the invoice key figure to the *Selection Details* area on the right of the screen.
4. Press *OK*. The help cell contains the description of the selection. The symbol indicates that a selection has been made for this help cell.
5. Next, choose *New Formula Cell* from the context menu of the required query cell in the upper screen area. The *Edit Formula* dialog box appears.
6. Enter a technical name for the exception cell. You can also enter a new description.
7. In the *Cells* group, double-click on the help cell defined on the "invoice" key figure, in order to select the cell.
8. Choose the buttons x 2.
9. Press *OK*. The cell now displays the exception cell whose formula is calculated instead of the cross product of the structure elements.

4.10.2.6 Defining Priorities

If you use two structures in a query, the system takes into account the properties, which have been specified (for the structures or for specific cells) as cross products of two structure elements. The system takes the properties into account according to the order in which the properties are specified. The last specification made has priority. On the *Priorities* tab, you can modify the priority handling by explicitly specifying which property is actually taken into account. This is a power user function.

Prerequisites

The query contains two structures and only one of these structures can contain key figures.

Context

The system automatically adds the *Priorities* tab.

Procedure

1. You are on the *Priorities* tab. Open the dropdown list for the field *Select Property* and select the property that you want to set.
2. Select a cell, whose values you want to change, in the row or column.
3. If you also want to change explicitly set cell values, deselect the field *Keep Explicit Cell Value*.

   The field *Keep Explicit Cell Value* is selected by default so that explicitly set cell values are retained.
4. Choose *Apply Row Value* to apply the default value to the row structure. A symbol indicates that the value has been applied (blue arrow pointing left).

5. Choose *Apply Column Value* to apply the default value to the column structure. A symbol indicates that the value has been applied (blue arrow pointing up).

Results

The changes to the values are displayed in the grid. If you have changed an explicitly set cell value, the system adds the expression **(Cell)** to the new value.

Next Steps

Save the query.

Related Information

Tab: Priorities [page 361]

4.10.2.6.1 Tab: Priorities

For every structure element or for a cell, you can explicitly specify which property has priority.

For the field *Select Property*, the system displays a list of properties, for which you can set priorities. These are display properties, which you can find in the definition of a structure element (selection or formula) or a cell on the *General* tab.

Here you see the two structures of the query with their structure elements in the grid. If the properties of a structure element have **not** been changed in the query definition, the system displays the default values configured on the InfoObject maintenance screen. However if a property of a structure element in the query definition has been explicitly changed, the system displays the changed value.

The values are indicated as follows in the grid:

Symbols in the grid

Symbol	Meaning
(equals sign in front of blue background)	The value in the column structure matches the value in the row structure.
(blue arrow pointing links)	The structure element inherits the property from the structure in the row.

Symbol	Meaning
(blue arrow pointing up)	The structure element inherits the property from the structure in the column.
	Symbol for the cell definition: An explicitly set cell value has priority. **Note** The field *Keep Explicit Cell Value* is selected by default, in order to save values in the event of any changes. This means explicitly set cell values are retained, even if the default value of the row structure or column structure is to be generally applied.

Related Information

Tab: General [page 330]
This link takes to you to the documentation for the properties of a selection.
Tab: General [page 346]
This link takes to you to the documentation for the properties of a formula.

4.10.2.7 Defining Filters

On the *Filter* tab, you can restrict characteristics or select exactly one key figure as a filter. This type of filter selection restricts the entire query. All the InfoProvider data is aggregated using the filter selection of the query.

You can define various filters:

- Filters with **Fixed Values**: The characteristic values selected as the filter selection **cannot** be changed at runtime.
- Filters with **Default Values**: The characteristic values selected as the filter selection are default values. The user can change or delete these values at runtime (in the preview for example).

Related Information

Filter [page 115]
Restricting Characteristics [page 363]
Specifying a Key Figure as a Filter [page 367]

4.10.2.7.1 Restricting Characteristics

On the *Filter* tab, you can restrict characteristics to characteristic values (or characteristic value intervals) or to hierarchy nodes (either fixed or variable).

Procedure

1. In the *BW Repository*, select a characteristic and drag and drop it onto one of the filter areas.
2. Open the InfoObject's context menu and choose *Restrict*. The *Edit Filter* screen opens on *Members* tab. The system shows the characteristic values (*members*) on the left of the screen.
3. Restrict the characteristic to:
    - a single characteristic value
    - a variable characteristic value interval
    - variable characteristic values
    - a specific hierarchy node
    - a variable hierarchy node
4. You can change the **sequence** of the selected characteristic values and variables in the filter:
    - To move a filter element up, choose *Up*.
    - To move a filter element down, choose *Down*.
5. Confirm your filter definition by pressing *OK*.

Related Information

Restricting Characteristics by Characteristic Values [page 363]
Restricting Characteristics on Hierarchy Nodes [page 365]

4.10.2.7.1.1 Restricting Characteristics by Characteristic Values

The following section explains how to restrict a characteristic to one or more characteristic values, or to one characteristic interval (fixed or variable).

Context

The filter contains a characteristic that you can restrict. The *Edit Filter* dialog box appears.

Procedure

1. If you want to select one or more **characteristic values** or a **characteristic interval**, proceed as follows:
   - If you want to **restrict the characteristic to a single characteristic value**, select the value and drag it to the filter definition area.
   - If you want to restrict the characteristic to a single characteristic value and give this a name, write this name in the *Direct Input* field and choose *Add*. The system validates your entry on the back end, and inserts the characteristic value with your name in the filter definition.
   - If you want to **restrict the characteristic to a characteristic range**, add the required start and end value of the range by double-clicking in the filter definition area, select the two characteristic values and choose *Create Range*.
2. If you want to restrict the characteristic to variable characteristic values rather than fixed values, choose a **characteristic value variable** as follows:
   a. Choose the *Variables* tab. On the left of the screen, the system shows a list of all variables. If you select a variable, the system shows the details of this variable in the lower part of the screen:
   b. Copy over the required variable by double-clicking on the filter definition area.
   c. You can specify offset values for a characteristic value variable (with the *Single Value* display range) by choosing *Create Range*. The *Enter Offsets Values for Range* dialog box appears. Here you can move the interval limits by adjusting the start value and end values (or just one of these values) by one integral offset value.

      You are using the **Current Calendar Year** variable in your filter definition. You want to select the last 5 years as the period for your data analysis. Select the variable, choose *Create Range*. The *Enter Offsets Values for Range* dialog appears. Set the lower limit to the offset value -5. Do not enter an offset (oder 0) for the upper limit. Press *OK*.
   d. If you want to define the range between two variables as a filter element, select the required start and end value of the range in the filter definition area and choose *Create Range*. If the variables are characteristic value variables (*Single Value*), you can also specify offset values for the interval built from these variables in the *Enter Offsets Values for Range* dialog box.
   e. If the variables are characteristic value variables with the *Single Value* display range, the *Enter Offsets Values for Range* dialog box opens.
3. The system applies the characteristic values or characteristic interval to the filter definition, in the order in which they are selected, using the operator **Equal** by default. You can change the **operation** to be used for the selected characteristic values and variables in the filter: Proceed as follows:
   a. Select a filter element.
   b. Press *Change Operator*. You can also choose this command in the filter element's context menu.
   c. Select the required operator. You can choose from the following operators:

Operator	Description
Equal To	Equal to
Greater Than or Equal To	Greater than or equal to
Greater Than	Greater than
Less Than or Equal To	Less than or equal to
Less Than	Less than

4. The system applies the characteristic values or the characteristic interval to the filter definition - using include by default. If there are values that you want to **exclude from the selection**, proceed as follows:

   - Select the filter element that you want to exclude and choose *Change Mode* > *Exclude*.
   - To include a filter element again, choose *Change Mode* > *Include*.

   > ⚠ Caution
   >
   > This function is only used with single values or single value variables.

Results

You have now restricted the characteristic to characteristic values or characteristic value intervals (fixed or variable).

If you have used variables, then before the query is executed, a dialog box appears where you can manually enter variable values (provided that the variable is flagged as input-ready).

Related Information

Variable [page 119]

4.10.2.7.1.2 Restricting Characteristics on Hierarchy Nodes

The following section explains how to restrict a characteristic to a hierarchy node (either a fixed or variable restriction).

Context

You have a characteristic in the filter, for which there are one or more hierarchies. You can restrict this characteristic to a hierarchy node. The *Edit Filter* dialog box appears.

The following table provides an overview of the various possibilities for restricting hierarchies and hierarchy nodes (fixed or variable restriction):

Using Variables for Hierarchies and Hierarchy Nodes

Selection	Example
Fixed Hierarchy Node - Fixed Hierarchy	Report on department A (fixed hierarchy node) for the hierarchy valid for 2015 (fixed hierarchy)

Selection	Example
Variable Hierarchy Node - Fixed Hierarchy	Report on chosen department (variable hierarchy node) for the hierarchy valid for 2015 (fixed hierarchy)
Variable Hierarchy Node - Variable Hierarchy	Report on a chosen department (variable hierarchy node) for the hierarchy valid for the current calendar year (variable hierarchy).

Procedure

1. If you want to restrict a hierarchy-bearing characteristic to a **specific hierarchy node**, proceed as follows:
   a. Choose the *Hierarchy Nodes* tab.
   b. Select a characteristic hierarchy using *Browse*. On the left of the screen, you see a list of all available hierarchy nodes.
   c. Add the hierarchy node of your choice by double-clicking it.
   d. Confirm by pressing *OK*.
2. If you want to allow a user to be able to restrict a characteristic at query runtime to a variable **hierarchy node** (of a hierarchy specified as fixed or variable), proceed as follows:
   a. Choose the *Hierarchy Nodes Variables* tab. In field *Referenced Hierarchy*, the system displays the first characteristic hierarchy that it finds as a reference hierarchy.
   b. If you want to use a different characteristic hierarchy as a reference hierarchy, choose *Change*. The *Select Hierarchy* dialog box appears: Choose *Browse* to select a different characteristic hierarchy. If you want to keep the hierarchy selection flexible, you can assign a variable for the reference hierarchy by using the variable symbol. Confirm your selection by pressing *OK*.
   c. Drag the hierarchy node variable into the right-hand screen area. The variable is either linked to the default reference hierarchy or to a reference hierarchy that you specified.

   The system applies the hierarchy node variable to the *Filter* and *Sheet* tabs.

   You have now restricted a characteristic to either a fixed hierarchy node of a fixed selected hierarchy, or to a variable node within a fixed or variable hierarchy. If you have used variables, then before the query is executed, a dialog box appears where you can manually enter variable values (provided that the variable is flagged as input-ready).

Related Information

Variable [page 119]

4.10.2.7.2 Specifying a Key Figure as a Filter

On the *Filter* tab, you can specify exactly one key figure as a filter if no key figure has been selected on the axles. The values of this key figure are displayed in the cells when the query is executed.

Procedure

1. In the *BW Repository*, select a key figure and drag and drop it onto one of the filter areas.
2. If you want to change the operator, select the required operator from the right screen area under *Details*.

Option	Description
Operator	Description
Equal	Equal to
LessThan	Less than
GreaterThan	Greater than
LessEqual	Less than or equal to
GreaterEqual	Greater than or equal to
ContainsPattern	Contains pattern

3. If you want to exclude the specified values, choose *Exclude*.

Example: Special Feature: Query Definition Without Key Figure

You want to define a query that only contains characteristics in the columns and in the rows. Every query must contain at least one key figure however. You can ensure this by adding a key figure to the *Filter: Fixed Values* and - if required - setting operators and excludes.

4.10.2.8 Defining Conditions

To achieve more effective data analysis, you can formulate conditions on the *Conditions* tab.

Context

In the results area of the query, the data is filtered according to the conditions so that only the part of the results area that you are interested in is displayed. You can restrict the view to the data of a query; with regards to defined threshold values or based on a defined ranked list.

Procedure

1. You are on the Conditions tab in the query definition. Open the context menu in *Conditions* and choose *Create Condition*. The system inserts a new condition.
2. Make the required settings on the *General* tab page.
3. Make the required settings on the *Assignments* tab page.
4. Press *OK*. You return to the query definition.
5. Save the query.

Results

You have defined or changed a condition for a query. Execute the query. Only objects that meet the defined conditions are displayed.

> ⓘ Note
>
> You can display the condition descriptions in the query view and activate or deactivate them there.

Related Information

Condition [page 116]
Tab: General [page 368]
Tab Page: Assignments [page 369]

4.10.2.8.1 Tab: General

On the *General* tab page, you can activate the condition and edit its parameters.

General

This area displays the default text for the *description*. You can change the description.

In the default setting, the new condition is flagged as *active*. If you change this setting, the condition will no longer be available for analysis of the query data.

The *Ignore NULL Values* flag is set by default. If the flag is set, NULL values are **ignored** and therefore never meet the condition (see Conditions Calculated in the SAP HANA [page 433]).

Condition Parameters

You can define the parameters of the condition here.

1. To create a parameter, choose *Add*. The *Define Condition Parameter* dialog box appears.
2. Select the structure element that you want the condition to be based on. If you are using two structures in your query, you can select a *First Member* and a *Second Member*.
3. Select the *comparison operator*.
   You can choose from the following operators for **threshold value conditions**:
   - *Equal*
   - *NotEqualTo*
   - *LessThan*
   - *GreaterThan*
   - *LessEqual*
   - *GreaterEqual*: >= Greater Than or Equal To
   - *Between*
   - *NotBetween*

   The values for these operators must be floating point numbers. For the operators *Between* and *Not Between*, you have to enter a value range by specifying an upper and a lower threshold value.
   You can choose from the following operators for **ranked list conditions**:
   - *TopN*
   - *BottomN*
   - *Top%*
   - *Bottom%*
   - *TopSUM*
   - *BottomSUM*

   For *Top/BottomN*, you have enter a positive integer. For *Top/Bottom%*, you have to enter a number between 0 and 100. For *Top/BottomSUM*, you can enter a decimal number (15.7 for example).
4. To remove a parameter, select it and choose *Remove*.

4.10.2.8.2 Tab Page: Assignments

One the *Assignments* tab page, you define a characteristic assignment that deviates from the default setting.

Mode

For the *Characteristic Assignments*, you have the following options:

- All Characteristics in drilldown *Independent*:
  This setting allows you to apply a condition in a general way. Depending on which characteristics you use in the drilldown, you obtain different results. This option is optimized for ranked list conditions, but can also be used for threshold value conditions with relative values.

> ⚠ Caution
>
> Characteristics that are already used in an active condition for characteristic combinations are not included.

- *Most Detailed Along Rows*:
  This option is optimized for threshold value conditions. The condition is applied to the most detailed characteristic of the rows.
- *Most Detailed Along Columns*:
  This option is optimized for threshold value conditions. The condition is applied to the most detailed characteristic of the columns.
- *Individual*:

Select Individual Characteristics

This setting allows you to evalute the condition for characteristics or characteristic combinations defined for specific drilldowns only. You can select any characteristic (of the characteristics used in the query in rows, columns, or in the free characteristics) or a characteristic combination.

- If you only want to evaluate the condition when a certain characteristic is present in the drilldown, select this characteristic.
- If you want to evaluate the condition for characteristic combinations, select multiple characteristics.

4.10.2.9 Defining Exceptions

On the *Exceptions* tab, you can define threshold values (exceptions) for a query. Data that deviates from these thresholds is highlighted in a different colour or flagged using icons. You can use these exceptions to instantly identify deviations from expected results.

Procedure

1. You are on the *Exceptions* tab in the query definition. Open the context menu in *Exceptions* and choose *Create Exception*. The system inserts a new exception.
2. Make the required settings on the *General* tab page.
3. Make the required settings on the *Definition* tab page.
4. Make the required settings on the *Cell Restrictions* tab page.
5. Press *OK*. You return to the query definition.
6. Save the query.

Related Information

Exception [page 117]
Tab: General [page 371]
Tab Page: Definition [page 372]
Tab Page: Cell Restriction [page 373]

4.10.2.9.1 Tab: General

On the *General* tab, you can activate the exception and edit its parameters.

General

This area displays the default text for the *description*. You can change the description.

In the default setting, the new exception is flagged as *active*. If you change this setting, the exception will no longer be available for analysis of the query data.

Exception Parameters

You can define the parameters of the exception here.

1. To create a parameter, choose *Add*. The *Define Exception Parameter* dialog box appears.
2. Select the *Alert Level*.
   Depending on the selection you make here, the system uses different background colors to highlight the values. The exception is displayed with the background color of the data cell or characteristic cell. The color shading ranges from dark green for alert level *Good1* through yellow for alert level *Critical* 1 to dark red for alert level *Bad3*. There are a total of nine color shades, representing nine different levels of priority.
   You can choose from the following alert levels:
   - *Good1*
   - *Good2*
   - *Good3*
   - *Critical1*
   - *Critical2*
   - *Critical3*
   - *Bad1*
   - *Bad2*
   - *Bad3*
3. Select the *comparison operator*.
   You can choose from the following operators:

- *Equal*: =
- *LessThan*:<
- *GreaterThan*: >
- *LessEqual*: <=
- *GreaterEqual*: >=
- *Between*: []
- *NotBetween*:][
- *ContainsPattern*

4. Enter a *value*. The values for most of these operators must be floating point numbers. For the operators *Between* and *Not Between*, you have to specfiy an upper and a lower threshold value in the *Value* and *To Value* fields.
   The input format depends on the country setting for the operating system.
5. To complete your parameter definition, choose OK.
6. To remove a parameter, select it and choose *Remove*.

4.10.2.9.2 Tab Page: Definition

On the *Definition* tab, you define on which structure element, at what moment in time and in which manner the exception should be applied.

Is defined on

You specify which key figures the exception is evaluated for. The exception can affect all structure elements (*All Elements*) of a structure or a given structure element (*Given Element*).

> ⓘ Note
>
> The exceptions in the query that is run can only take effect if the key figure specified here is contained in the drilldown.
>
> If say you define an exception on a key figure K1 (Billed Quantity for example), that affects key figure K2 (Net Sales for example), it is not enough if just K2 is in the drilldown. The exception can only take effect if K1 is in the drilldown as well.

Time of Evaluation

You can set whether the evaluation of the exception is to take place **before** or **after** a local calculation (see Tab: Calculations [page 333]).

In the standard setting, the exception is evaluated **before** a local calculation (*Before List Calculation*).

Is Displayed in Data Cells

You can define whether the exception affects data cells or characteristic cells. In accordance with your setting, the highlighting for the exception is displayed on the numbers for the selected key figure or the text for the most detailed characteristic value. The standard setting here is *Is Displayed in Data Cells*.

When you display the exception on data cells, you can choose whether the exception is to be displayed on the *evaluated structure element*, on a *given element*, or on *all structure elements*.

You can use the option *Active* to specify whether the affected setting should be applied (activated).

Display in Characteristic Cells

When you display the exception on characteristic cells, you can choose whether the exception is to be displayed on *rows*, on *columns*, or on both *rows and columns*.

You can use the option *Active* to specify whether the affected setting should be applied (activated).

4.10.2.9.3 Tab Page: Cell Restriction

On the *Cell Restriction* tab, you specify which cells the exception is to be evaluated for.

Standard operator for all characteristics not listed

Here, you specify whether the exception should affect *all rows* or *only result rows* in the characteristics that you have not defined a restrictoin for in the lower screen area.

Cell Restrictions

By choosing *Add*, you can create a cell restriction for a characteristic regarding the exception. You can change a cell restriction by choosing *Edit* and delete it by choosing *Remove*.

You can choose from any of the free characteristics, or characteristics from the rows and columns used in a query. You cannot use characteristics that are already used in another restriction row in the list.

Select the required validity areas (*operators*) for the characteristic and restrict the characteristics as required.

Possible Validity Areas

Validity Area	Description
Everything	The validity area is not restricted. The exception is valid for all drilldown states for the characteristic.
Totals Only	The exception is valid for aggregated values of the characteristic only.
Everything Except Totals	The exception applies to all values of the characteristic apart from aggregated values.
Fixed Values	The exception is valid for just one particular value of the characteristic (for example Spare Parts under Product Group). Select the required characteristic value.
Level	The exception is valid for a particular hierarchy level of the characteristic only. Select the required hierarchy level.

4.10.2.10 Saving Queries

Once you have finished defining a query, you can save it under its current name by choosing *File* > *Save*.

Context

You can also save a query if you have not made any changes to the definition. You can save your query at any time.

4.10.2.11 Publishing to Roles

If you want to make sure that users with particular role authorizations can see a particular query, you can assign this query to the corresponding roles.

Prerequisites

You are authorized to perform role maintenance.

Assignment to Role

1. You are in the query. Choose *Publish to Role* from the toolbar. The screen for maintaining role assignments appears.
2. Select the required roles listed under *Selection options* and move these to the *Assignments* section by drag and drop or by choosing the corresponding entry in the context menu. If you would like to assign further queries to the same roles, you can copy your selection to the buffer.
3. Save your role assignment.

Test

To test your role assignment, open the context menu for your project and choose *Open Query by Role*.

4.10.2.12 Precalculating Value Sets

A precalculated value set can be used as a filter in an analytic query. The filter is defined as a variable of type *predefined value set* and is executed at runtime. If the precalculated value set is large, query runtime can be improved considerably. A precalculated value set is always assigned to a characteristic that is used in the query.

Context

There are three different ways of entering precalculated value sets:

- Using an analytic query
- Entering them manually
- By referencing a table

The following section describes entry using a query.

The precalculated value set with query assignment is stored in a database table.

Procedure

1. You use a special query to determine the set of characteristic values to be available as a precalculated value set.
2. You use a variable to make this value set available in other queries.
3. You define a query for your reporting. In this query you use the variable in the filter, and can thus switch between various precalculated value sets at query runtime.

4.10.2.12.1 Filling a Value Set

You fill a value set by defining a query with a filter that restricts the amount of characteristic values in accordance with your requirements.

Procedure

1. Define a query for filling the value set. Create the characteristic that you want to create a precalculated value set for in the rows.

   > → Tip
   >
   > We recommend making the query as simple as possible. The focus should be on selecting the characteristic values for filling the value set.

2. Define a filter that restricts the characteristic values at runtime in accordance with your requirements.
3. Choose *Update Value Set by Query* in your query's toolbar. The *Precalculated Value Sets (Buckets)* dialog box appears. The system displays the value sets belonging to the characteristic in the rows.
4. Select the target value set.

   > ⓘ Note
   >
   > If you choose the *Show All Query-Related Value Sets* option, the system displays all value sets with a query assignment, regardless of the characteristic.

   > ⓘ Note
   >
   > From the context menu for the entries in this results list, you can create additional values set or delete existing ones.

5. Choose *Fill Value Set*. The system writes the selected characteristic values to the value set.

Example

1. You define a query by dragging characteristic CITY to Rows. The system automatically adds it to *Filter: Default Values*. The columns contain your key figure structure, with key figure SALES for example.
2. You use characteristic COUNTRY in the fixed filter and restrict it to the value Germany.
3. Choose *Update Value Set by Query* in your query's toolbar. The *Precalculated Value Sets (Buckets)* dialog box appears. The system displays the value sets belonging to characteristic CITY.
4. Select the target value set. Let's call it CITY STANDARD SET.
5. Choose *Fill Value Set*. The system writes the selected characteristic values - all city names stored for Germany that is - to value set CITY STANDARD SET.
6. Then create another value set containing only the cities in the German state of Baden-Württemberg, called CITY BW SET.

4.10.2.12.2 Defining a Variable for a Precalculated Value Set

To access a precalculated value set at query runtime, you first define a variable that represents the precalculated value set.

Prerequisites

You have used a special query to determine the amount of characteristic values to be available as a precalculated value set.

Procedure

1. Create a characteristic value variable with processing type *Manual Input / Default Value* (*User Entry*) for the same target characteristic that you used in the query for filling the value set.
2. For the *Var. Represents* field, choose the *Precalculation* (*Precalculated*) display area. The variable now represents the precalculated value set for the required target characteristic.
3. Open the Variables Editor. We advise against setting the *Input Type* to *Optional* on the *General* tab page. We recommend setting it to *Mandatory with Initial* instead. The variable will then be visible at query runtime for the query to be used.
4. You have the option of defining a precalculated value set as a default value set on the *Default Values* tab page.

> ⚠ Caution
>
> Note that you will receive a list of all available value sets from the Content Assistant CTRL + SPACE, not just those that relate to the characteristic of your variables.

Example

You create a characteristic value variable called CITY VALUE SET VARIABLE, make all of the settings described above, and set CITY STANDARD SET as the default value set on the *Default Values* tab page.

Related Information

Display Area [page 139]

4.10.2.12.3 Using Precalculated Value Sets

In this topic you will learn how to use a precalculated value set by means of a variable in the filter.

Prerequisites

You have used a special query to determine the amount of characteristic values to be available as a precalculated value set.

You have used a variable to make this value set available in other queries.

Procedure

1. Create a query in which you use the previously defined variable as a filter.

   > ⓘ Note
   >
   > During execution of the query, the values from the precalculated value set are used as a filter.

2. At runtime, you can switch between various value sets. To do this, choose *Browse* under *Variables* and select the value set defined for the target characteristic.

Results

By using precalculated value sets, you do not need to define filter values. This improves runtime, especially in complex scenarios.

Example

1. You define a second query in which you include characteristic CITY in the fixed filter and restrict to variable CITY VALUE SET VARIABLE.
2. At runtime, you can switch between the value sets CITY STANDARD SET and CITY BW SET, thereby viewing the figures for all German cities in the one, and for cities in the state of Baden-Württemberg in the other.

4.10.2.13 Using Reusable Query Components

You can reuse objects that appear in the *BW Repository* under *Reusable Components* in all the queries of an InfoProvider. You can also reuse characteristic variables.

When changes are made to a reusable object, note that these are applied everywhere that the object is used.

You can use the BW search to find an overview of the reusable query components, which are used in a project or available system-wide.

1. Choose *Search* > *Search...* in the user menu. As an alternative, you can use the shortcut `Ctrl + H` or press the *Search* () button in the *BW Modeling* perspective.
2. A popup opens. Go to the *BW Object Search* tab page.
3. Enter a search string * (any string).
4. Choose *Advanced Filter* the object type ELEM [query element] as an additional restriction for the search. The system displays all reusable query elements under *Subtypes* (except the query itself).
5. Select the required subtype and choose *Search* to start the search.

Related Information

Defining Calculated Key Figures [page 379]
Defining Restricted Key Figures [page 382]
Defining the Structure [page 385]
Defining Filters [page 387]
Defining Variables [page 389]

4.10.2.13.1 Defining Calculated Key Figures

If you recalculate a key figure of an InfoProvider using a formula, then this calculated key figure can be reused in all queries of this InfoProvider.

Procedure

1. Open the context menu for the InfoProvider and choose *New* > *Calculated Key Figure*. The *New Calculated Key Figure* wizard opens. The system applies the properties from the BW project and the InfoProvider, which the project is based on, as the default settings.
2. Enter a description and a technical name for the new calculated key figure.
3. If you have created a directory structure for your reusable objects in the component hierarchy maintenance screen (transaction code RSZTREE), you can enter the required folder in field *Folder*. Choose *Browse*. A dialog box appears where you can select an object. (You can also leave this selection blank.

You can also leave this blank. Then the system displays the calculated key figure in folder *Reusable Components* > *Calculated Key Figures*).

4. You have now configured the basic settings. Choose *Finish* to create the calculated key figure. The editor appears.
5. In the *Formula* area on the *General* tab, you will find the input field for the formula, which you want to use to calculate the key figure. Base key figures, restricted key figures, already calculated key figures and formulas can be used as operands. The procedure for defining a reusable calculated key figure correspond to the procedure for defining a formula.
6. Change the settings as required for the properties of the calculated key figure in the *Properties* screen area.

   The *Properties* screen area contains the following tabs, which correspond to the properties of a formula:

   - *General* tab: For calculated key figures, the same properties are available in the *Formula Collision* and *Display* screen areas as are available for the properties of a formula. The only difference is that the *Formula* screen area is **not** available because the calculated key figure **cannot** be changed in the query.
   - *Aggregation* tab
   - *Calculations* tab
7. Select the calculated key figure.

Results

You have created a calculated key figure for an InfoProvider. This key figure can be reused in queries, which are based on this InfoProvider.

You can find the calculated key figure either in the *BW Repository* of the query under the *Reusable Components* > *Calculated Key Figures*, or by using the search. To search for the key figure, find the toolbar in the *BW Modeling* perspective and choose (*Search*). Now you can search for the query on the *BW Object Search* tab. In the *Advanced Filter* area, you can restrict the search to the type ELEM [Query Element] and subtype CKF [Calculated Key Figure].

Related Information

Using Calculated Key Figures [page 381]
Overwriting Calculated Key Figures Locally [page 381]
Defining a Formula [page 344]
The following links take you to the documentation for related procedures.
Tab: General [page 330]
Tab Page: Aggregation [page 349]
Tab: Calculations [page 333]
The following links take you to the documentation for properties of the formula.
Calculated Key Figure [page 118]
This link takes to you to the documentation for concepts.

4.10.2.13.1.1 Using Calculated Key Figures

You can reuse objects that appear in the *BW Repository* under *Reusable Components* in all the queries of an InfoProvider.

Context

You are on the *Sheet Definition* tab in the query definition.

Procedure

1. Select the required calculated key figure and add it to the query definition. There are various options here:
   - In the *BW Repository*, select one or more objects from the available reusable query components under *Calculated Key Figures*. Drag and drop the objects onto the relevant area of the query definition.
   - Open the context menu for the area of the query definition and choose *Add Reusable Components* > *Calculated Key Figures*. A dialog box appears where you can select the required object. Select one or more calculated key figures. Choose *OK* to add them to the query definition.
2. Save your query definition.

4.10.2.13.1.2 Overwriting Calculated Key Figures Locally

You can overwrite a calculated key figure for a query. However, the changes that you make remain local.

Context

You are in the definition of a query that contains the calculated key figure. You want to further restrict this calculated key figure (by using an additional characteristic selection, for example) and change other properties on the *General*, *Conversion* and *Calculations* tabs.

> **Note**
>
> Note that you reusable key figures within a query *cannot* be overwritten. A local query element is created that references the calculated key figure. You can now perform further modifications on this local query element.

Procedure

1. Select the calculated key figure and choose *Edit* from the context menu. The *Edit Selection* dialog box appears.
2. Enter a technical name and a description.

   > **Note**
   >
   > Note that the technical name only relates to the local query element and not to the calculated key figure in use.

3. Make changes to the definition of the calculated query as required.
4. Save the query.

Results

You have locally overwritten a calculated key figure in your query.

You will find the calculated key figure either in the query definition or in the outline view of the query elements. In the *Calculated Key Figure* directory, you can see the base reusable calculated key figure and the locally overwritten calculated key figure in the *Sheet* area of the query definition. Both key figures have the same technical key but have different UIDs.

4.10.2.13.2 Defining Restricted Key Figures

You can restrict the key figures of an InfoProvider by using one or more characteristic selections. The key figure that is restricted by one or more characteristic selections can be a basic key figure, a calculated key figure, or a key figure that is already restricted. A restricted key figure is reusable

Procedure

1. Open the context menu for the InfoProvider and choose *New* > *Restricted Key Figure*. The *New Restricted Key Figure* wizard opens. The system applies the properties from the BW project and the InfoProvider, which the project is based on, as the default settings.
2. Enter a description and a technical name for the restricted key figure.
3. In the *Key Figure* field, you can enter the *Key Figure* of the InfoProvider. Choose *Browse*. A dialog box appears where you can select an object. (You can also leave this blank and create the selection in the editor).
4. If you have created a directory structure for your reusable objects in the component hierarchy maintenance screen (transaction code RSZTREE), you can enter the required folder in field *Folder*. Choose *Browse*. A dialog box appears where you can select an object. (You can also leave this blank. Then the system displays the calculated key figure in folder *Reusable Components* > *Restricted Key Figures*).

5. You have now configured the basic settings. Choose *Finish* to create the restricted key figure. The editor appears.
6. On the *General* tab, you can configure the required restrictions in the *Selection* area. To do this, drag and drop the required key figure of the InfoProvider into the selection area and restrict the key figure by selecting one or more characteristic values.

   > **Note**
   >
   > You can also use variables instead of characteristic values. Note that, due to technical reasons, you **cannot** use variables with processing type *Replacement Path* with replacement from a query in restricted key figures.

7. Change the settings as required for the properties of the restricted key figure in the *Properties* screen area.

   The *Properties* screen area contains the following tabs, which correspond to the properties of a selection:

   - *General* tab: For calculated key figures, the same properties are available in the *Constant Selection* and *Display* screen areas as are available for the properties of a selection. The only difference is that the *Selection* screen area is **not** available because the calculated key figure **cannot** be changed in the query.
   - *Conversion* tab
   - *Calculation* tab

8. Select the restricted key figure.

Results

You have created a restricted key figure for an InfoProvider. This key figure can be reused in queries, which are based on this InfoProvider. The restricted key figure is not an actual element in this query. It is a placeholder that references the restricted key figure created for the InfoProvider.

You can find the restricted key figure either in the *BW Repository* of the query under the *Reusable Components* > *Restricted Key Figures*, or by using the search. To search for the key figure, find the toolbar in the *BW Modeling* perspective and choose (*Search*). Now you can search for the query on the *BW Object Search* tab. In the *Advanced Filter* area, you can restrict the search to the type ELEM [Query Element] and subtype RKF [Restricted Key Figure] .

Related Information

Using Restricted Key Figures [page 384]
Overwriting Restricted Key Figures Locally [page 384]
Defining Selections [page 329]

The following links take you to the documentation for related procedures.

Tab: General [page 330]
Tab: Conversion [page 333]
Tab: Calculations [page 333]

The following links take you to the documentation for properties of the selection.

Restricted Key Figure [page 118]

This link takes to you to the documentation for concepts.

4.10.2.13.2.1 Using Restricted Key Figures

You can reuse objects that appear in the *BW Repository* under *Reusable Components* in all the queries of an InfoProvider.

Context

You are on the *Sheet Definition* tab in the query definition.

Procedure

1. Select the required restricted key figure and add it to the query definition. There are various options here:
   - In the *BW Repository*, select one or more objects from the available reusable query components under *Calculated Key Figures*. Drag and drop the objects onto the relevant area of the query definition.
   - Open the context menu for the area of the query definition and choose *Add Reusable Components* > *Restricted Key Figures*. A dialog box appears where you can select the required object. Select one or more restricted key figures. Choose *OK* to add them to the query definition.
2. Save your query definition.

4.10.2.13.2.2 Overwriting Restricted Key Figures Locally

You can overwrite a restricted key figure for a query. However, the changes that you make remain local.

Context

You are in the definition of a query that contains the restricted key figure. You want to further restrict this restricted key figure (by using an additional characteristic selection, for example).

> **Note**
>
> Note that you reusable restricted key figures within a query *cannot* be overwritten. A local query element is created that references the restricted key figure. You can now perform further modifications on this local query element.

Procedure

1. Select the restricted key figure and choose *Edit* from the context menu. The *Edit Selection* dialog box appears.
2. Enter a technical name and a description.

   > **Note**
   >
   > Note that the technical name only relates to the local query element and not to the restricted key figure in use.

3. Make changes to the definition of the restricted query as required.
4. Save the query.

Results

You have locally overwritten a restricted key figure in your query.

You will find the restricted key figure either in the query definition or in the outline view of the query elements. In the *Restricted Key Figure* directory, you can see the base reusable restricted key figure and the locally overwritten calculated key figure in the *Sheet* area of the query definition. Both key figures have the same technical key but have different UIDs.

4.10.2.13.3 Defining the Structure

When you create a structure for an InfoProvider, it can be reused in all queries for this InfoProvider.

Procedure

1. Open the context menu for the InfoProvider in question and choose *New* > *Structure*. The *New Structure* wizard for creating reusable objects opens. The system applies the properties from the BW project and the InfoProvider which the project is based on as the default settings.
2. Enter a description and a technical name for the structure.
3. You have now configured the basic settings. Choose *Finish* to create the structure. The editor appears.
4. You can change properties and select structure members here.
   - Under *General*, you can change the technical name and the description.
   - Under *Display*, you can make settings regarding *Visibility of Structure Member* and *Zero Suppression*.
   - Under *Details*, you can select *members*. To do this, choose one of the following options in the context menu:
     - *New Selection*

- *New Formula*
- *Add Key Figures...*
- *Add Reusable Components*

5. Save your structure.

Related Information

Using Structures [page 386]
Locally Overwriting a Structure [page 387]
The following links take you to the documentation for related procedures.
Structure Properties [page 327]
This link takes to you to the documentation for the properties of a structure.
Structure [page 119]
Structure [page 112]
This link takes to you to the documentation for concepts.

4.10.2.13.3.1 Using Structures

You can reuse objects that appear in the *BW Repository* under *Reusable Components* in all the queries of an InfoProvider.

Context

You are on the *Sheet Definition* tab in the query definition.

Procedure

1. Select the object you require from the query components that can be reused in all queries for an InfoProvider under *Reusable Components* > *Structures*.
2. Drag the object to the area of your choosing in the query definition.

Related Information

Locally Overwriting a Structure [page 387]

4.10.2.13.3.2 Locally Overwriting a Structure

A reusable structure is a global object that **cannot** be changed in the context of a query. You can make a local object from this kind of structure however.

Procedure

1. If you want to make changes to the structure in the context of your query, choose *Dereference* from the context menu.

   The system creates a local copy of the structure.
2. Enter a new description.
3. Make the required changes to the local structure object.

Results

You have used the global structure object as the basis for a local structure object. You can use this structure in the context of your query, but not in other queries.

Related Information

Defining Structure Elements [page 328]

4.10.2.13.4 Defining Filters

When you create a filter for an InfoProvider, it can be reused in all queries for this InfoProvider.

Procedure

1. Open the context menu for the InfoProvider in question and choose *New* > *Filter*. The *New Filter* wizard for creating reusable objects opens. The system applies the properties from the BW project and the InfoProvider which the project is based on as the default settings.
2. Enter a description and a technical name for the filter.
3. You have now configured the basic settings. Choose *Finish* to create the filter. The editor appears.
4. You can select the screen areas *Fixed Values* and *Default Values* in the editor.
5. Save your filter.

Related Information

Reusing Filters [page 388]
Overwriting Filters Locally [page 389]
The following links take you to the documentation for related procedures.
Filter [page 119]
Filter [page 115]
This link takes to you to the documentation for concepts.

4.10.2.13.4.1 Reusing Filters

You can reuse objects that appear in the *BW Repository* under *Reusable Components* in all the queries of an InfoProvider.

Context

You are on the *Filter* tab in the query definition.

Procedure

1. Select the object you require from the query components that can be reused in all queries for an InfoProvider under *Reusable Components* > *Filter*.
2. Drag the object to the area of your choosing in the query definition.

Related Information

Overwriting Filters Locally [page 389]

4.10.2.13.4.2 Overwriting Filters Locally

A reusable filter is a global object that **cannot** be changed in the context of a query. You can make a local object from this kind of filter however.

Procedure

1. If you want to make changes to the filter in the context of your query, choose *Dereference* from the context menu.

   The system creates a local copy of the filter.
2. Enter a new description.
3. Make the required changes to the local filter object.

Results

You have used the global filter object as the basis for a local filter object. You can use this filter in the context of your query, but not in other queries.

Related Information

Defining Filters [page 362]

4.10.2.13.5 Defining Variables

You can define reusable variables for InfoObjects. These variables will also be available for filtering queries.

Context

The basic properties of the variable (such as variable type, processing type, reference characteristic, display area and input type), which you set when you create the variable, **cannot** be changed later on. However, you can edit the technical name, description and specific settings for the processing type in the properties of the variable later on.

Procedure

1. Open the context menu of the required InfoObject and choose *New* > *Variable*. The *New Variable* wizard opens. The system has applied the properties from the BW project and the reference characteristic as the default setting.
2. Enter a description and a technical name for the variable.
3. Select the variable type for field *Type of Variable*.
4. Select the processing type for field *Processing By*. Depending on the variable type, only certain processing types are useful. The Variables Wizard automatically only offers you the processing types that are possible for your selection.
5. If you want to specify a different InfoObject as a reference characteristic, choose *Browse* for field *Ref. Characteristic*.
6. Choose the display area for the *Var. Represents* field.
7. If you have selected processing type *Replacement Path*, choose *Next*. In the next step you specify the **Replacement Rule**.
8. Once you have finished defining the variable, choose *Finish*. The system creates the variable. The Variables Editor opens and you see the screen area that shows the properties of the variable.
9. In the properties of the variable you can change various settings as required on the *General* tab. You can also, for example, set default values or settings for starting the query on the available tabs (tab availability depends on processing type).
10. To delete a variable, choose the delete symbol (*Delete*) in the properties of the variable.

Results

You have created a variable, which can be used in queries containing the reference characteristic.

You can find the variable either in the *BW Repository* of the query under the reference characteristic, or by using the search. To search for the variable, find the toolbar in the *BW Modeling* perspective and choose (*Search*). Now you can search for the query on the *BW Object Search* tab. Under *Advanced Filter*, you can restrict the search to the type ELEM [Query Element] and subtype VAR [Variable].

You can also use the variable in InfoArea *(0BW4_GLOBAL) Global Objects*.

Related Information

Variable [page 119]
Overview of Variable Types and Processing Types [page 120]
Variable Type [page 121]
Processing Type [page 122]
Display Area [page 139]
The following links take you to the documentation for variable concepts.
Example: Using a Variable with Replacement from Query [page 391]

Variable Properties [page 393]
Restricting Characteristics [page 363]
Restricting Characteristics by Characteristic Values [page 363]
Restricting Characteristics on Hierarchy Nodes [page 365]
The following links take you to the documentation for relevant procedures.

4.10.2.13.5.1 Example: Using a Variable with Replacement from Query

Context

You want to insert the result of the query **Top 5 Products** as a variable into the query **Sales - Calendar Year/Month**.

Procedure

1. Select the characteristic **Product** and choose *New Formula* from the context menu. The *New BW Variable* wizard opens.
2. Enter a description and a technical name for the new characteristic value variable.
3. Select variable type *Characteristic Value* for field *Type of Variable*.
4. Select processing type *Replacement Path* for field *Processing By*.
5. Choose *Next*.
6. Choose *Query* as the **Replacement Rule**.
7. Choose the query **Top 5 Products** using the input help.
8. You have not completed the definition of the variable. Choose *Finish*. The system creates the variable. The Variables Editor opens and you see the screen area that shows the properties of the variable.

Results

You can now insert the variable into query **Sales - Calendar Year/Month**. This allows you to determine how the sales for these five top-selling products has developed month for month.

> **Note**
>
> Every time you execute query **Sales - Calendar Year/Month**, query **Top 5 Products** is executed beforehand (hence the term "pre-query"). This can lead to longer runtimes. If you do not need to immediately execute the pre-query, we recommend that you use a variable, which represents a precalculated value set. This

could be the case for example if you do not need to determine the top 5 products every time, because they only change on a monthly basis.

Related Information

Using Precalculated Value Sets (Buckets) [page 605]

4.10.2.13.5.2 Using a Variable with Replacement from BRFplus

In the *BW Modeling Tools* , you assign a a BRFplus function that you define in the *Business Rules Framework plus* to a variable with processing type *BRFplus-Exit*.

Prerequisites

In *Business Rules Framework plus* (transaction BRFPLUS), you have defined the following:

- A suitable BRFplus application
- A BRFplus function (as described in BRFplus Exit [page 136])

> **Note**
>
> Program RSR_BRFPLUS_TOOL supports you in your work with variables in the BRFplus functions that can be used by variables in SAP BW/4HANA.

Function	Description
Display BRFplus Function for Var.	With this option, the system displays the available BRFplus functions that can besed for a variable on the basis of their context signature. Double-click to call the BRFplus maintenance screen.
Execute Variable	You can use this option to execute a BRFplus function from this report. Note that not all information that can be used as parameters at runtime is available here. Also note that the first execution of a BRFplus function subsequent to the logic being changed always results in re-generation of the function and therefore takes longer than subsequent execution runs.

Function	Description
Create BRFplus Function	With this option, you can create new BRFplus functions with a suitable context signature. In the *Business Rules Framework plus*, you then have to define the required logic.

Procedure

1. Variables of processing type BRFplus Exit have a *BRFplus Function* tab page. On this tab page, you can assign a BRFplus function to the variable. This BRFplus function is then executed when the variable is processed.
2. You can set the required logic for this kind of BRFplus function in the *Business Rules Framework plus* (transaction BRFPLUS).

Related Information

BRFplus Exit [page 136]

4.10.2.13.5.3 Variable Properties

If you have created a variable using the Variables Wizard, the properties of the variable appear in the right-hand area of the Variables Editor.

This area is divided into tabs. The individual tabs and settings vary, depending on the variable type and processing type. The Variables Editor automatically displays only those settings that are relevant and available.

Related Information

Tab: General [page 394]
Additional Tabs Available for Specific Processing Types [page 395]

4.10.2.13.5.3.1 Tab: General

The *General* tab of the Variables Editor is required in almost all variable types. Only variables with processing type *Replacement Path* do not require this tab. In this case, the fields on this tab are not input-ready.

Screen Area: General

The system displays the technical name and the description. You can change the description as required.

Screen Area: Global Settings

The system displays the reference characteristic, the variable type and the processing type. Choose *Reference Characteristic* to open the properties of the reference characteristic.

Screen Area: Details

The system shows the display area of the variables as not changeable.

The following properties are changeable:

- *Input Type*:
  - *Optional*: If you select this setting, the variable does not have to be filled with a value at runtime.

    > ⓘ Note
    >
    > Variables with the processing type *Authorization* have the default setting *Optional*.

  - *Mandatory With Initial*: If you select this setting, at least one value has to be specified for the variable at runtime. The initial value (#) is permitted. The initial value # means "unassigned", meaning that you can use it to explicitly select all data records where this characteristic does not have any instances.
  - *Mandatory Without Initial* If you select this setting, at least one value has to be specified for the variable at runtime. The initial value (#) is **not** permitted. You must enter one or more concrete values to be able to execute the query.
- *Input-Ready*:
  Using this option, you can specify whether the variable is input-ready when the query is executed. The option is activated by default.
  - If a variable is input-ready, then when the query is executed, the system first shows the requested values, on which you can confirm or modify the default values.
  - If you deactivate the option *Input-Ready*, the system automatically executes the query with the default value.
  - If you deactivate this option *Input-Ready* but you have specified the variable value as mandatory and you have not specified a default value, then the system **cannot** execute the query. When you try to execute the query, an error message appears.

- *Copy Personalization from Variable*:
  Users can personalize variable values for input-ready variables. The personalized values are saved for each variable and user.If you want to use the same personalization data in more than one variable, enter the names of the variables that the personalization data is stored under in this input field. Enter the names manually, or select them using the input help.

  > ⓘ Note
  >
  > You have to enter the variable without an ampersand (&).

  > ✤ Example
  >
  > You have defined a variable "Customer1", which is mandatory and input-ready. You now define a "Customer2" variable, which is mandatory (initial value not allowed) and input-ready. By entering variable "Customer1" in the input-ready field for variable "Customer" choosing *Copy Personalization from Variable*, you ensure that both variables have the same personalized values.

Related Information

Display Area [page 139]
This link takes to you to the documentation for variable concepts.
Additional Tabs Available for Specific Processing Types [page 395]
This link takes to you to the documentation for variable properties.

4.10.2.13.5.3.2 Additional Tabs Available for Specific Processing Types

The tabs *Default Values*, *Replacement Path* and *Currencies and Units* are available in the variable editor - depending on the processing type.

Default Values

This tab is available for variables of processing types *User Entry* and *Text*.

In the *Default Values* screen area, you can enter one or more default values depending on the variable type. These appear when the query is executed.

> ⓘ Note
>
> If the variable is input-enabled, you can overwrite the default value and choose another one when executing the query. If the variable is **not** input-enabled, the query is filled with the default value. In this case, you can **no longer** change the value when executing the query.

- With characteristic variables you can enter one or more default values, depending on the selected processing type. Choose *Browse*, *Add* or *Edit* to open a dialog box where you can select default values.
- Only one default value is required for formula and text variables. Enter the required default value into the input field manually.
- With hierarchy variables and hierarchy node variables, choose *Change* to open a dialog box where you can select a specific hierarchy or a specific hierarchy node as a default value.

Replacement Path

The tab is available for all variable types with processing type *Replacement Path*.

In the *Replacement Rule* screen area, the system shows the source which variable replacement is based on (CURRENT MEMBER variable, query, variable or InfoObject). This replacement rule cannot be modified.

You can still change the object that you want to use for the replacement however. Choose *Browse*. A dialog box appears where you can select a different object.

Depending on the replacement type, you can set additional properties in the *Replacement Rule* and *Details* screen areas.

Currencies and Units

This tab is available for variables of *Formula* variable type.

Formula variables usually represent numbers without dimensions. In the *Dimension Settings* screen area, you can select a dimension indicator, which means that the formula variable represents a specific dimension. Depending on the dimension, you can then determine a currency or a unit.

The following table provides an overview of the various available dimensions:

Dimension indicator

Dimension	Currency and unit
Amount	Currency
Quantity	Unit
Price	You can select a currency and a unit. (Price = Currency/Unit)
Number	You do not need to make any more entries.
Date	You do not need to make any more entries.
Time	You do not need to make any more entries.

Related Information

Overview of Variable Types and Processing Types [page 120]
Replacement Path [page 123]
Replacement from Characteristic Value [page 124]
Replacement from CURRENT MEMBER Variable [page 130]
Replacement from Query [page 130]
Replacement from Variable [page 131]

4.10.2.14 Query Properties (General Tab)

You can make various settings for your query in the *General* tab.

The *General* tab is divided into the following screen areas:

Related Information

Screen Area: General [page 397]
Screen Area: Output Settings [page 398]
Screen Area: Remote Access [page 400]
Screen Area: Variables Order [page 400]
Screen Area: Result Location [page 400]
Screen Area: Zero Suppression [page 401]
Screen Area: Universal Display Hierarchy [page 401]
Screen Area "Extended" [page 403]

4.10.2.14.1 Screen Area: General

In the *General* screen area of the *General* tab, you can view the general properties of the query and configure various settings.

The following general properties are available:

- *Technical Name*: Displays the technical name of the query.
- *Description*: You can enter or change the query description here.
- *InfoProvider*: Displays the technical name of the InfoProvider that the query is based on. Choose *InfoProvider* to open the maintenance screen for the InfoProvider.
- *Key Date*: Every query has a key date. For time-dependent data, the key date determines the time for which the data is selected. The default value for the key date is the date when the query is executed: <today>.
  1. To enter a key date manually, choose *Change...*. The *Edit Key Date* dialog box appears.

2. Select a date from the calendar.
   If you select 01/01/1999 for example, the system reads time-dependent data for 01/01/1999.
3. Choose OK.
4. Choose  to switch between the default value and manual input for a key date.

4.10.2.14.2 Screen Area: Output Settings

In the *Output Settings* screen area of the *General* tab, you can view and configure settings for query output.

The following settings are available for query output:

- *Adjust Formatting after Refreshing*: This setting allows you to decide whether the formatting of the query is adapted after refreshing.
  The formatting for the query underlines the type of data contained in a cell and highlights the structure of the delivered results. The position and size of the cell area in the analysis grid change when the query is refreshed. The *Adjust Formatting After Refreshing* function guarantees that the format is adjusted when the updated query data is refreshed. The setting *Adjust Formatting after Refreshing* is activated by default.
- *Suppress Repeated Key Values*: You can specify whether to display identical key values more than once in the query. The *Suppress Repeated Key Values* setting is active in the default setting. This means that only the first key value is displayed for each characteristic, while other identical key values are hidden.

> ⓘ Note
>
> The key value is the value that the characteristic is presented with. This can be the characteristic description or the characteristic key for example. You can specify how the characteristic is presented by using characteristic properties.

> → Tip
>
> In the following example, the *Suppress Repeated Key Values* setting is activated:

Region	Order	Sales
DE	A	EUR 14,900
	B	EUR 15,800
	C	EUR 17,600
USA	A	EUR 12,800
	C	EUR 32,800

In the following example, the *Suppress Repeated Key Values* setting is deactivated:

Region	Order	Sales
DE	A	EUR 14,900

Region	Order	Sales
DE	B	EUR 15,800
DE	C	EUR 17,600
USA	A	EUR 12,800
USA	C	EUR 32,800

- *Show Scaling for Measures*: Key figures can have scaling factors and units or currencies. If you activate the *Show Scaling for Measures* setting, an additional header row is added to the row or column header in the query. This additional header row contains information about the scaling factor and the unit or currency, as long as it is consistent (for example EUR 1,000).
  This setting is normally deactivated.

  → Tip

  In the following example, the settings *Suppress Repeated Key Values* and *Show Scaling for Measures* are activated:

Region	Order	Sales
		EUR 1,000
DE	A	14
	B	15
	C	17
F	D	12
	E	32

- *Sign Format*: This allows you to specify how the minus sign (-) is displayed. You have the following options for displaying negative values:
  - *Before* the number: -123.45. The minus sign is positioned before the value (this is the default setting).
  - *After* the number: 123.45-. The minus sign is positioned after the value.
  - *In brackets*: (123.45). Negative values are displayed in brackets.
- *Zero Format*: You have the following options for displaying zeroes:
  - *With Currency*: Zero with currency/unit. Zeroes are displayed with the relevant currency/unit, for example EUR 0.00. This is the default setting.
  - *Without Currency*: Zero without currency/unit. Zeros are displayed without a currency or unit entry, for example 0.00.
  - *As Space*: Zero as an empty space. Cells containing a zero value remain empty.
  - *Custom*: Zero as a custom text. If you choose this setting, the Show Zeroes field is activated. You can now enter the required value (character, number, or letter), for example an asterisk (*). All cells that contain a zero will now be filled with this value.

4.10.2.14.3 Screen Area: Remote Access

In the *Remote Access* screen area of the *General* tab, you can view and configure the settings for remote access.

The following remote access options are available:

External reporting tools use queries as their data source. This allows you to determine whether and how the query is accessed.

- *By OData*: The integration of SAP BW/4HANA with SAP Gateway allows you to provide analytic queries of SAP BW/4HANA as OData queries for mobile scenarios. Queries with OData access are specialized for REST access. To generate the query as an OData query, select *By OData*.
- *By OLE DB for OLAP*: If you want to release this query as the data source for external reporting tools that communicate using interface OLE DB for OLAP, select *By OLE DB for OLAP*.

  > ⓘ Note
  >
  > Queries that contain formulas with the operators %RT, %CT, %GT, SUMRT, SUMCT, SUMGT, and LEAF **cannot** be released for OLE DB for OLAP. The required result can be achieved using constant selection.

- *External SAP HANA View*: If the query is based on an InfoProvider that supports SAP HANA views, you can use this setting to make the system create a SAP HANA view for the current query.

4.10.2.14.4 Screen Area: Variables Order

In the *Variables Order* screen area of the *General* tab, you can view and configure settings for the input order of variables in the query.

If your query contains variables, this setting allows you to define the order in which the user enters them. Choose *Up* and *Down* to move the variables up or down. When the query is executed, the system generates a variable input window, which displays the variables in the order specified here.

4.10.2.14.5 Screen Area: Result Location

In the *Result Location* of the *General* tab, you can specify where the result is displayed.

- *Rows*: above or below (*Above/Below*)
- *Columns*: left or right (*Left/Right*)

In the default setting, the results are displayed on the bottom right.

4.10.2.14.6 Screen Area: Zero Suppression

Under *Zero Suppression* on the *General* tab, you can define whether or not to display rows or columns containing zero values.

Under *Applies to*, you specify whether suppression of zeros should be applied to rows and columns, only to rows, or only to columns.

The following options (*Condition*) are available for suppressing zeros:

- *Only result value must be zero*: If the rows and columns contain characteristics, all rows or columns with a result of zero are **not** displayed.

  > ⓘ Note
  >
  > The row or column contains the values 1, -1, 1, -1 for example, meaning that the result is 0. If you select *Only result value must be zero*, the entire row or column is **not** displayed.

- *All values must be zero*: Columns or rows containing zero values in **all** cells are **not** displayed.

4.10.2.14.7 Screen Area: Universal Display Hierarchy

In the *Universal Display Hierarchy* screen area of the *General* tab, you can specify whether characteristics are displayed hierarchically in the rows or columns, and up to which level in this display hierarchy data is read. This can be significant in terms of performance.

Note that the display hierarchy always relates to the current query definition.

> ✤ Example
>
> - For example, if you have the characteristics country, federal state and city in the rows and you do **not** select the option *Universal Display Hierarchy*, the system reads all the data for all characteristics in the drilldown when the query is executed.
>
>   Without Universal Display Hierarchy

Country	Federal State	City	Sales
A	AA	AAA	100
		AAB	100
	Total		200
B	BA	BBA	50
		BBB	50
		BBC	100
	Total		200
C	CA	CCA	50
	Total		50

Country	Federal State	City	Sales
Total			450

- However, if you choose the option *Universal Display Hierarchy* and specify when the query is executed to only expand **up to level 1** (in other words, country), the system only reads the data on the country level (sales per country and not sales per city) when the query is executed.

  With universal display hierarchy up to level 1

Country	Sales
A	200
B	200
C	50
Total	450

- However, if you choose the option *Universal Display Hierarchy* and specify when the query is executed to only expand **up to level 2** (in other words, federal state), the system only reads the data on the federal state level, when the query is executed, and displays the data in a hierarchy.

  With universal display hierarchy up to level 2

Country	Sales
A	
AA	200
B	
BA	50
BB	150
C	
CA	50
Total	450

- If you insert a different characteristic (at the first position above the characteristic Country, for example), such as the characteristic Payment Method, and you have activated the option *Universal Display Hierarchy* up to level 2, the system displays the data by payment method and then underneath by country.

  Changed query definition with universal display hierarchy up to level 2

Payment Method	Sales
BZ01	
B	150
C	50
BZ02	

Payment Method	Sales
A	200
B	50
Total	450

If you want to view the data by country and by federal state on the executed query, expand the relevant nodes of the universal display hierarchy. The system reads the required data again.

4.10.2.14.8 Screen Area "Extended"

Under **Extended** in the **General** tab, you can make data integrity settings. These are relevant if your back-end system supports this.

Request-Status

If the InfoProvider is suitable for this, you can specify how recent the displayed InfoProvider data is. In one InfoProvider, data can be read with different request statuses. The following request statuses are possible:

- Green: Request loaded successfully
- Yellow: Request not loaded successfully (still loading)
- Red: Request not loaded successfully

The system supports the following alternatives for InfoProvider data integrity:

Options for Specifying Data Integrity

Option	Description
All InfoProviders up to released status (RQTS0): All InfoProviders Up to Released Status (RQTS0) (0)	The system reads all data up to the first open request or request with errors. This is the default setting for queries.
Planning-Mode-Support-Enabled ADSOs up to current version: Data Mart DataStore objects and direct update DataStore objects up to the current version (1)	In addition to the **All InfoProviders up to released status (RQTS0)** mode, the system still reads the data of a plannable DataStore object up to and including the first open request. **Note** The **InfoProvider up to current version** is suitable for queries that are defined for example on a CompositeProvider that contains plan data in a DataStore object and actual data in another InfoProvider.

Option	Description
All InfoProviders up to current version (to Qualok): All InfoProviders Up to Current Version (to Qualok) (2)	In addition to the **All InfoProviders up to released status (RQTS0)** mode, the system also reads the data up to and including the first open request for all InfoProviders.
All InfoProviders up to current version (all green requests): All InfoProviders Up to Current Version (All Green Requests) (3)	The system reads all requests that have no errors (all green requests); this includes requests created after requests with errors or requests that were not completed.
Every InfoProvider in CompositeProvider according to its setting: Every InfoProvider in the CompositeProvider according to its setting (7)	The system reads the data from all InfoProviders of a CompositeProvider according to their request statuses. This setting is only possible for CompositeProviders.
All InfoProviders (including red requests): All InfoProviders Complete (Red Requests as Well) (9)	The system reads all available data; all incorrect requests and all open requests for a query are included.

If you do not want to select a particular request status, you can also define a variable.

Cold Store (Nearline)

If the query is based on an InfoProvider that data has already been stored for in a cold store, you can stipulate that the stored data is read and displayed when the query is run.

Cold Store (Nearline) Access Options

Option	Description
Use Cold Store According to Basic Provider Setting: Cold store access is made in accordance with the InfoProvider setting.	This option is only relevant for a CompositeProvider. If this option is activated, the setting for each individual InfoProvider in the CompositeProvider is taken into account.
Do Not Read Cold Store: Cold store access is switched off	Even if data has been stored in a cold store, the query does not access this data. If a CompositeProvider contains an InfoProvider that allows access, this setting overwrites this and prevents access. **Note** If you have deactivated access in the InfoProvider settings, you **cannot** activate it again in the query properties.

Option	Description
Read Cold Store: Cold store access is switched on	Access to data in the cold store is possible if one of the following conditions are met: • There is a data archiving process for the InfoProvider. • This is connected to a cold store. • The selection conditions of the navigation step intersect with the previously archived data area. In a CompositeProvider, this setting allows access to all the InfoProviders of the CompositeProvider that meet the aforementioned conditions (even if access is deactivated in the InfoProvider).

You can also define a variable for cold store access.

Document Links for:

If you select the corresponding setting, the system displays links to documents - provided that you have created the documents for the InfoProvider data, master data or metadata of the query.

4.10.2.15 Dependency Structure Tab

On the *Dependency Structure* tab, you can see a graphic visualization of the components used in the query.

The display of the query components makes it simple for you to clarify questions such as:

- Which dependencies are there between components?
- How can changes to individual components affect the query result?
- Which query objects must be transported?

Using the context menu, you can print out the display, highlight dependencies or export the entire display or part of the display as a .bmp file.

4.10.2.16 Changing Query Runtime Properties (Runtime Properties tab)

On the *Runtime Properties* tab, you can make settings for individual analytic queries that deviate from the settings for the InfoProvider that the queries are based on.

Context

The *Runtime Propertis* tab is divided into the following screen areas:

- *Common Runtime Properties*
- *Runtime Profile Properties*

> ⚠ Caution
>
> Note that the system only displays settings that are relevant for the query in question.
>
> Example: If the query does not have any virtual key figures or characteristics, settings that relate to virtual key figures or characteristics are **not** displayed.

Procedure

1. To be able to change one of the query properties, first switch via  from the default value set on the InfoProvider to manual entry of query properties.
2. Make the required settings for the query or queries in question. This can be either activation or deactivation of the query, or selection of a particular value.

   Note that the other runtime profile properties you can specify depend on the chosen data integrity profile.
3. Save the query.

Related Information

General Runtime Properties [page 407]
Runtime Profile Properties [page 412]

4.10.2.16.1 General Runtime Properties

On the *Runtime Properties* tab, you can make general runtime settings for individual analytic queries that deviate from the settings for the InfoProvider that the queries are based on.

- *Process Key Figure with High Precision*
- *No Parallel Processing*
- Calculating Commutative Formulas After Aggregation (*Calc. Commut. Formulas After Aggregation*)
- *Query is used as InfoProvider*
- *Generation Log*
- *OLAP Effort*

Related Information

Calculating Key Figures with High Accuracy [page 407]
Not Using Parallel Processing [page 408]
Calculating Commutative Formulas After Aggregation [page 409]
Query as InfoProvider [page 410]
Generation Log [page 411]
OLAP Effort [page 411]

4.10.2.16.1.1 Calculating Key Figures with High Accuracy

By setting a specific accuracy, you decide which number format is used internally for calculating key figures. The level of calculation accuracy can be specified in the InfoObject maintenance transaction for an individual key figure (system-wide), and specified in the query properties for all key figures of a particular query.

Use

Decimal floating point numbers are used for calculation in the analytic manager. This avoids problematic conversions of the data type float from the decimal system to the dual system and the other way round.

There are two levels of precision for decimal floating point numbers:

- Decimal floating point numbers with 16 places ("decfloat16", "short decfloat")
- Decimal floating point numbers with 34 places ("decfloat16", "long decfloat")

Short decfloats are used by default in the analytic manager for numerical data, to minimize the memory requirements for the main memory and the OLAP cache. If you choose the query property *Process Key Figure with High Precision*, the Analytic Manager calculates **all** key figures of the selected query with 34 figure decimal floating point numbers (long decfloats).

→ Tip

Choose this option for application areas where a high accuracy is required (for queries in the accounting and cost accounting areas for example).

⚠ Restriction

The processing effort for long and short decfloats is similar. Due to the higher memory requirements for long decfloats, we recommend that you do not use this number format for all query key figures. Only use it for key figures that require a high degree of accuracy. You can do this by using the calculation precision setting for the required key figures in the InfoObject maintenance screen.

4.10.2.16.1.2 Not Using Parallel Processing

A query based on an InfoProvider is split internally into subqueries. These are automatically processed in parallel. You can use this property to switch off parallel processing for a query.

Use

In most cases, parallel processing of subqueries improves performance. In certain cases however, sequential processing can achieve the best improvements in query performance:

If you choose the *Do Not Use Parallel Processing* property, the selected query will be processed sequentially from now on.

Related Information

Processing Queries [page 408]

4.10.2.16.1.2.1 Processing Queries

Use

A query can be divided into sub-queries by the system. This division is performed in the following circumstances:

- The query contains cells with a constant selection or other definitions that cannot be read using a logical read operation.
- The query contains complex selections, such as calculated or restricted key figures with hierarchy conditions. The selections are then split into multiple accesses.

If dividing the query results in more than one sub-query, the read operation is performed in parallel by in the default setting.

Parallel Processing

Multiple dialog work processes are required in order to execute queries in parallel. The maximum degree of parallelism determines the maximum number of work processes that are used for each query.

The actual degree to which queries are executed in parallel depends on the load on the system at any given time and lies between 1 (sequential processing) and the maximum value. If the number of sub-queries is greater than the maximum level of parallelism, all existing sub-queries are divided between the work processes determined by the degree of parallelism.

The results of all sub-queries are collected at a synchronization point and collated to form an overall result.

In sequential processing, the sub-queries are processed one after the other. The interim result is immediately passed on to the analytic manager.

Sequential Processing

In the following cases, the system chooses sequential (non-parallel) processing:

- The entire query consists of one sub-access only.
- The query is running in a batch process.
- The query requests non-cumulative key figures.
- There are not enough dialog processes available when the query is executed. These are required for parallel processing.
- The query is configured for non-parallel processing.
- You want to save the result of the query in a file or a table.

Partial Parallel Processing

Queries on an InfoProvider with non-cumulative key figures cannot be processed in parallel because of their high memory consumption. All queries that are based on this kind of InfoProvider are therefore divided into sub-queries that are separated from the other sub-queries. The ‘normal’ sub-queries are processed in parallel first, and the sub-queries that are based on the non-cumulatives are processed sequentially afterwards.

4.10.2.16.1.3 Calculating Commutative Formulas After Aggregation

In query monitor, you can use the *Query Properties* dialog box to specify when a commutative formula is calculated for existing queries.

Context

Commutative formulas are can be calculated before or after aggregation with the same result. By default, the OLAP Engine calculates commutative formulas, which use formula exception aggregation, before aggregation.

A formula calculated **before** aggregation is part of the generated query report and can be executed with higher system performance than a formula calculated after aggregation (which must be evaluated at runtime). The

result of a formula before aggregation can be saved in the OLAP cache. The formula before aggregation is calculated for every data record, which is transferred from the InfoProvider to the OLAP Engine (see statistics event ID 9010: total number transportable records, DBTRANS).

If the formula is calculated **after** aggregation, then the number of calculations depends on the granularity of the drop-down characteristic and the reference characteristic of the formula exception aggregation.

The use of function *Calculate Commutative Formulas After Aggregation* can be more efficient if data aggregation is high due to the standard aggregation and exception aggregation. This means that only a few aggregated data records remain, for which the formula needs to be calculated.

4.10.2.16.1.4 Query as InfoProvider

It can sometimes make sense to process query data as an InfoProvider: The advantage in doing this is that you can calculate the query's functions in the Analytic Server **before** reading the data. This improves performance, especially in the case of simple queries that contain a very large amount of data.

> ⚠ Caution
>
> If you use a query as an InfoProvider, not all query functions are supported.

The following functions are supported:

- Standard aggregations (SUM, MIN, MAX)
- Formulas that are totaled before aggregation (+, -, constant factor)
- Unit and currency translations
- Non-cumulative key figures
- Constant selection
- Formulas with quotients, like prices
- Formulas with exception aggregation

In the following cases, it is not possible to activate a query as an InfoProvider:

- The query has input-ready required variables.
- Temporal hierarchy join is enabled.
- The query has more than one structure.
- The technical name of the query is more than 20 characters in length.

The following settings are not taken over from the query:

- Set hierarchies
- Local aggregations
- Elimination of internal business volume
- Conditions

Special Features in the Result of the Query Execution

When you execute a query, the result is displayed differently, according to whether or not you use the query as an InfoProvider. If you use a query as your InfoProvider, note the following:

- For values with a mixed currency or unit basis (10€ + 5$ for example) and for values with compound units or currencies (15 $/kg for example), the unit display in the result is the initial (unassigned) value.
- Key figure values where nothing (NULL) is displayed in the result set display the value "0".

4.10.2.16.1.5 Generation Log

You can use this property to specify whether a log should be written when a query is regenerated.

Context

This function is used for test and support purposes. In certain cases, queries have to be generated again; for example, when a query definition is changed. If you select the *Generation Log* option, the system writes entries to the application log. These entries provide information on why the query was generated again.

4.10.2.16.1.6 OLAP Effort

If InfoProviders store large amounts of data, certain queries can retrieve large result sets. **OLAP effort** is the maximum effort expected by Analytic Manager for processing the data it retrieves.

Use

This property can force the use of filters or the removal of drilldown characteristics if queries exceed the specified upper limit for OLAP effort. This restricts the effort for a query and with this the waiting time for the query result.

You can decide what has priority here: Would you like to restrict the query effort or would you prefer to have a result in spite of longer processing and response times?

The effort for Analytic Manager depends on the size of the result set and on the complexity of the query:

- **Complex Queries**: Although a large data set is retrieved from the database in the back end, the result set that the user sees is quite small in some cases. This occurs in the case of exception aggregations and currency translations for example.
- **Simple Queries**: Queries without complex functions or with functions that have been moved out to the SAP HANA database are considered simple queries.

Overview of the modes

- **Default (0):**
  A default value of the user's choosing can be set for the query. Normally the default value is "infinite". This therefore has no effect on the queries.
- **Low (4):**
  This mode is suitable for simple queries with small result sets.
- **High (6):**
  This mode is suitable for simple queries with large result sets or for complex queries with small or medium-sized result sets.
- **Medium (5):**
  This mode is suitable for simple and complex queries if the size of the result set is situated between low and high.
- **Infinite (9):**
  This mode returns the query result regardless of the size of the OLAP effort.

4.10.2.16.2 Runtime Profile Properties

On the *Runtime Properties* tab, you can make runtime settings for individual analytic queries that deviate from the settings for the InfoProvider that the queries are based on. The *Data Integrity Profile* property has a significant influence on the other selection options.

- *Data Integrity Profile*
- *Query Read Mode*
- *Query Cache Mode*
- *Cache Usage Mode*
- *Update Cache Objects in Delta Process*
- *Type of SP Grouping for Delta Caching*
- *Operations in SAP HANA/BWA*
- *Materialize Intermediate Query Result*
- *Use Selection of Structure Elements*
- *Also Read Child Members*
- *Optimization Mode*

- *Optimization Periods*
- *Stat. Detail Level*

Related Information

Data Integrity Profile [page 413]
Read Mode [page 415]
Cache Mode [page 416]
Cache Usage Mode [page 417]
Delta Caching [page 418]
Specifying the InfoProvider Grouping for Delta Caching [page 418]
Operations in SAP HANA [page 419]
Materializing the Intermediate Result of the Query [page 438]
Using the Selection of Structure Elements [page 440]
Reading Child Members with Hierarchy Structure Nodes [page 441]
Optimization Mode [page 442]
Optimization Periods [page 443]
Statistic Details Display [page 443]

4.10.2.16.2.1 Data Integrity Profile

The data integrity profile provides you with an easy way of testing the data integrity of a query, without the need to make separate expert data integrity or request status settings in the read mode, cache mode, cache usage mode and the InfoProvider grouping.

You can choose from the following profiles:

- **' ': Standard: No Profile (expert settings are used)**
  The behaviour of the query is tested using the expert settings.
- **'G': Guaranteed Stable Data from Start of Query**
  We only recommend this mode for queries with very few characteristics and restricted navigation, as it can take up a large amount of memory on the BW application server. It can even cause the query to terminate. If you choose this profile, the query will always show the data from when the query starts. When called for the first time, the system transfers a data snapshot of the query - which is requried for all possible drilldowns or navigation - to a memory segment of the BW application server. Every subsequent navigation step in the query is only performed on the transferred data set.
  Note that the initial reply time can be very long, as a snapshot of the data has to be created when the query starts. The system uses read mode*Query to Read All Data at Once (A)*.

  > ⚠ Restriction
  >
  > The following restrictions apply:
  >
  > - It is **not** possible to add free characteristics in this mode.
  > - It is **not** possible to use *Operations in SAP HANA / BWA* in this mode (with the exception of *Optimized Access*).

- Whenever variables belonging to the query's static filter are changed, the system creates a new snapshot.

- **'S': Stable Data from Start of Query**
  This mode offers stable data with adequate performance.
  Data mart DataStore objects are read with data integrity mode (or request status) *All InfoProviders Up to Released Status (RQTS0)*. If InfoProviders cannot read old datasets, navigation steps could fail, and it might be necessary to set the *Operations in SAP HANA* option to *Optimized Access* mode. The system then changes the read mode from *Query to read when you navigate or expand hierarchies (H)* to *Query to read during navigation (X)*.
- **'P': Provider specific: Plannable (or transactional) InfoProviders with current data**
  This mode implicitly offers the standard behaviour for the system, which is always applied if no expert settings are changed.
  InfoProviders that can read old datasets are read the same way as in profile 'S'.
  Plannable (or transactional) InfoProviders are read with data integrity mode (or request status) *Real-Time InfoCubes up to Current Version*. For plannable (or transactional) InfoProviders and InfoProvider that cannot read old datasets, cached data is used if available. If not, current data must be read. The *Operations in SAP HANA* option is **not** adjusted. If InfoProviders cannot read old datasets, the read is changed from from *Query to read when you navigate or expand hierarchies (H)* to *Query to read during navigation (X)*.
- **'C': Consistent Current Data**
  This mode returns current data every time a new list is calculated. Operations that cause a new list to be calculated are all operations that affect the entire list. These include filters, changing characteristic properties, and changing the settings for row and column axese.
  Whenever hiearchy nodes are expanded or collapsed, the system keeps new numbers consistent with the numbers that are already in the list.
  Data mart DataStore objects are read with data integrity mode (or request status) *All InfoProviders Up to Current Status (Qualok)*. Plannable (or transactional) InfoProviders wth current data are read with data integrity mode (or request status) *Real-Time InfoCubes up to Current Version*. If InfoProviders cannot read old datasets, this can cause the system to change the *Operations in SAP HANA / BWA* option to *Optimized Access* mode, and to change the read mode from *Query to read when you navigate or expand hierarchies (H)* to *Query to read during navigation (X)*.
- **'M': Most Current Data**
  This mode always displays the most current data and does not keep the data consistent in a list displayed if you drill down in the list.
  Data mart DataStore objects (including plannable ones) are read with data integrity mode (or request status) *All InfoProviders Complete (Red Requests as Well)*. The *Operations in SAP HANA / BWA* option can be used without restrictions.
- **'D': Dirty Data**
  This mode behaves the same as 'M', but deactivates the cache in the Analytic Manager. The reply time can therefore be very long.

> **Note**
>
> All InfoProviders that use the delta cache can read old datasets.
>
> These are mainly data mart DataStore objects or providers created by customers, which implement delta cache interface IF_RSD_DELTACHACE_SUPPORT with the corresponding settings.

4.10.2.16.2.2 Read Mode

The read mode determines how the OLAP processor acquires data during navigation. You set the mode for InfoProviders in Customizing and for queries in the query monitor.

The following types are supported:

- **Query to be read when you navigate or expand hierarchies (H)**
  The amount of data transferred from the database to the OLAP processor is the smallest in this mode. However, it has the highest number of read processes.
  In the *Query to Read Data during Navigation* mode, the data for the fully expanded hierarchy is requested for a hierarchy drilldown. In the *Query to be read when you navigate or expand hierarchies* mode, the data across the hierarchy is aggregated and transferred to the OLAP processor on the hierarchy level that is the lowest in the start list. When expanding a hierarchy node, the children of this node are then read.
- **Query to read during navigation (X)**
  The OLAP processor only requests data that is needed for each navigational status of the query in the front end. The data that is needed is read for each step in the navigation.
  Unlike with the *Query to be read when you navigate or expand hierarchies* mode, presentation hierarchies are always imported completely at leaf level here.
  The OLAP processor can read data from the main memory when the nodes are expanded.
  When accessing the database, the best aggregate table is used and, if possible, data is aggregated in the database.
- **Query to read all data at once (A)**
  There is only one read process in this mode. When you execute the query on the front end, all data in the main memory area of the OLAP processor that is needed for all possible navigational steps of this query is read. During navigation, all new navigational states are aggregated and calculated from the data from the main memory.

The *Query to be read when you navigate or expand hierarchies* read mode significantly improves performance in almost all cases compared to the other two modes. The reason for this is that only the data the user wants to see is requested in this mode.

Compared to the *Query to be read when you navigate or expand hierarchies* mode, the *Query to read data during navigation* setting only effects performance for queries with presentation hierarchies.

Unlike the other two modes, the *Query to Read All Data At Once* setting also improves performance for queries with free characteristics. The OLAP processor aggregates on the corresponding query view. For this reason, the aggregation concept, working with pre-aggregated data, is least supported in the *Query to Read All Data At Once* mode.

We recommend using the *Query to be read when you navigate or expand hierarchies* mode.

Only choose a different read mode in exceptional circumstances. The *Query to Read All Data At Once* read mode might be of use in the following cases:

- The InfoProvider does not support selection. The OLAP processor reads significantly more data than the query needs anyway.
- A user exit is active in a query. This prevents data from already being aggregated in the database.

4.10.2.16.2.3 Cache Mode

Definition

The cache mode defines whether and how the query results and navigational states calculated by the OLAP processor should be saved as highly compressed data in a cache.

The mode can be set as the default value for an InfoProvider in Customizing and for a query in the query monitor.

Use

Caching is a means of improving query performance. Care should be taken to balance benefits against cost however:

- Benefits: Better query response time
- Cost: Expenditure involved in structuring and storing data in the cache

You can store cache data in the database or locally at runtime. The option that you choose depends on various parameters. These include:

- How often the query is requested
  We recommend saving frequently requested queries in the cache. There are practically no limitations on the memory space available in the database for the database cache.
- The complexity of the query
  Caching improves performance for queries whose evaluation is more complex. We recommend keeping complex data processed by the OLAP processor in the cache.
- How often data is loaded
  The cache does not constitute an advantage if query-relevant data is frequently changed and therefore has to be loaded frequently, since the cache has to be regenerated every time.

Overview of Cache Modes

- **Cache mode I: Cache is inactive**
  New data is read during every single navigation step.
- **Cache mode 0: Local cache**
  The database cache is deactivated. In this case, the system works exclusively with the local cache.
- **Cache mode D: Cache in database**
  The cache data is persistent in database tables. In this mode, no data is displaced, and the memory size is unlimited. The database instance for the table carries the load. More space is required, but this method also saves time.
- **Cache mode P: Every provider as set in the InfoProvider definition**
  This mode is only available for queries on MultiProviders and CompositeProviders. It makes it possible to cache the data from the contained InfoProviders in accordance with the cache made set in the definition of each individual InfoProvider.

4.10.2.16.2.4 Cache Usage Mode

The cache usage mode is an expert tool for the cache mode. It defines when cache entries can be used to answer a query, and when the data has to be read again.

Cache Usage Modes

The following cache usage modes are available:

- **"S"** Always display data from query start (**s**table data)
  The time that the query starts at determines the status of the data. If you navigate within the query, the data is always displayed in the status it had when the query started.

  > ⓘ Note
  >
  > If a provider is not able to provide an old version of the data, and the requested data is not in the query cache, the system displays a warning.

- **"C"** Always display current data (**c**urrent data)
  Current data is taken during every navigation step. This does not mean that data is read from the database during every navigation step however. If the status of the InfoProvider has not changed, and the cache already contains data to resond to the current query, the system uses this data. The data only needs to be read from the database again if the status of the InfoProvider has changed.

  > ⓘ Note
  >
  > Cache usage mode "Curent Data" makes it possible to always take new data for InfoProviders that are updated frequently.

- **" "** Daten from query start if possible, otherwise current data (standard behaviour)
  In the case of InfoProviders that offer a mechanism for reading older versions of the data, the time when the query starts determines the status of the data, as with the "Stable Data" mode. InfoProviders that do not facilitate reading of older versions of the data provide new data if the data cannot be taken from the cache.

InfoProvider-Specific Information

- **Data Mart DataStore Objects:**
  The behavior of these DataStore objects can generally be defined with the semantics for data integrity, as InfoCubes offer the option of reading older data versions.
  With cache usage mode "C - Current Data", it is also possible using DataStore objects with the data integrity mode (or request status) *All InfoProviders Up to Released Status (RQTS0) (0)* to make changes in the released status visible with every navigation step.

4.10.2.16.2.5 Delta Caching

You can use the *Update Cache Objects in Delta Process* property to specify that individual data in a query is added to the cache.

Use

With the delta cache procedure, you do not need to read all data for the query from the database when the data basis changes. Whether the data is read directly from the database or from the cache does not have any impact on the query results.

Data mart DataStore objects are suited to the delta cache process because their request ID can be used to identify which entries have been added.

> ⚠ Caution
>
> The cache still has to be rebuilt when:
>
> - changes are made to metadata, master data, hierarchies, or exchange rates
> - InfoProviders are not suited to the delta cache process
> - InfoProviders have been compressed

4.10.2.16.2.6 Specifying the InfoProvider Grouping for Delta Caching

You can use the *Type of SP Grouping for Delta Caching* setting to optimize the behaviour of the delta cache.

Use

If a query is based on a CompositeProvider, you can specify that the data that each of the InfoProviders for the CompositeProvider supplies is to be treated separately and stored separately in the cache.

The advantage of this grouping is that you do not have to read all data for all InfoProviders again when data for just one sub-InfoProvider changes.

The disadvantage of this grouping however is that it requires more memory space because the data cannot be aggregated using the various InfoProviders. The data is not aggregated until the query is run.

> ⚠ Caution
>
> The grouping is only taken into account if the delta cache process is activated for the query.

The following modes are supported:

- **No Provider Partitioning (0)**
  There is no partitioning. All results for the InfoProvider are merged into one result and cached. If the data for an InfoProvider changes, the cache has to be rebuilt. This is the default setting.
- **Grouping depending on InfoProvider Types (1)**
  With grouping based on the InfoProvider types, all InfoCubes are saved in a group. The remaining InfoProviders are either stored in one group or stored in multiple groups according to their properties. DataStore objects or InfoProviders that can supply hierarchy nodes are grouped for example.
- Every Provider Separate (3)
  With this setting, all InfoProviders are stored separately.

4.10.2.16.2.7 Operations in SAP HANA

The system always checks whether optimized operations can be performed in the SAP HANA for the InfoProvider that the selected query is based on.

If you are using a CompositeProvider, the system checks each InfoProvider it contains.

The following standard modes are available:

- *Defensive (J)*: This mode corresponds to expert mode *Optimized Access (3)*.
- *Standard (M)*: The **optimizer for operations in SAP HANA** decides whether the query is executed with expert mode *Formulas Calculated in SAP HANA with Complex Currency or Unit (8)* or expert mode, *Optimized Access (3)*.
- *Offensive (P)*: The **optimizer for operations in SAP HANA** decides whether the query is executed with expert mode *Conditions Calculated in SAP HANA (9)* or expert mode, *Optimized Access (3)*.

For experts, you also have the option of selecting one of the following expert modes provided they have been activated.

> ⓘ Note
>
> For information about activation, see 2505732
>
> Note that the **optimizer for operations in SAP HANA** is **not** called when you choose one of these expert modes.
>
> Also note the following case: A user activates the switch for expert modes and sets an expert mode for a query. After this, other users who use this query can also see the selected expert mode even if they have not activated the switch for expert modes.

- *Expert: No optimized operations in SAP HANA (0)*
  If you do **not** want the system to use optimized operations in theSAP HANA database, choose this mode.
- *Expert: Individual Access per InfoProvider(2)*:
  The SAP HANA is used with this mode in the same way as with the **Optimized Access (3)** mode.
  For CompositeProviders however, a separate access is made for each participating InfoProvider.

  > ⚠ Caution
  >
  > If the mode has been changed manually, this manual setting can still be used - regardless of whether this was done in the query or directly in the InfoProvider.

- *Expert: Optimized Access (3)*:
  In this mode, optimized access to the SAP HANA database is activated.

  > **Note**
  >
  > If virtual characteristics or key figures are used in the query, a warning appears.

  For CompositeProviders, joint optimized access is made to all participating InfoProviders.
- *Expert: Exception Aggregation (6)*
  You can calculate exception aggregations in the SAP HANAdatabase.

  > **Note**
  >
  > If it is not possible to use optimized operations for some of the InfoProviders contained in a CompositeProvider or MultiProvider, the system reads these InfoProviders without optimized access and temporarily copies the data read in them to the SAP HANA database.

  If the exception aggregation **cannot** be calculated in SAP HANA because of the functionality used, the *Optimized Access (3)* mode is used.
- *Expert: Formulas Calculated in SAP HANA (7)*:
  This mode allows you to calculate formulas with exception aggregations in the SAP HANA. This requires SAP HANA 1.0 SP11 (or higher). All prerequisites that apply for mode *Exception Aggregation (6)* also apply here. If such a formula operand of a formula does not have an exception aggregation, this operand is also performed in. SAP HANA
- *Expert: Formulas Calculated in SAP HANA with Complex Currency or Unit of Measure (8)*
  In addition to mode *Formulas Calculated in SAP HANA (7)*, the system also supports formulas with a complex currency or unit. This requires SAP HANA 1.0 SP12 Revision 122.04 (or higher).
- *Expert: Conditions Calculated in SAP HANA (9)*:
  The system supports the calculation of conditions. This requires SAP HANA 2.0 SP3 Revision 30 (or higher).

Related Information

Optimizer for Operations in SAP HANA [page 421]
Learning Program for the Optimization of Operations in SAP HANA [page 422]
Exception Aggregation in SAP HANA [page 425]
Formulas Calculated in SAP HANA [page 428]
Formulas Calculated in SAP HANAwith Complex Currency or Unit of Measure [page 431]
Conditions Calculated in the SAP HANA [page 433]

4.10.2.16.2.7.1 Optimizer for Operations in SAP HANA

The optimizer helps you to select the relevant mode in the system that can execute operations to your advantage SAP HANA

If you execute a query with standard mode, the optimizer decides whether the query is executed with the *Optimized Access (3)* mode (negative decision) or with export mode *Formulas Calculated in SAP HANA with Complex Currency or Unit of Measure (8)* (positive decision).

If an enhanced expert mode is used, you get a smaller result set in the database (DBTRANS). This shortens the processing time of the result set in the analytic manager. On the other hand, there are additional setup costs for enhanced expert modes. For this reason, it is only useful to execute a query with an enhanced expert mode if the benefit (shorter processing time for the result set returned by the database) is greater than the additional setup costs.

The following table illustrates this:

Example Query	**Execution in SAP HANA**		**No Execution in SAP HANA**	
Response Time (ms)	DBTRANS	Response Time (ms)	DBTRANS	
Q01	5093	2	7015	76092
Q02	5766	12	3000	10

Q01 is faster when executing operations in SAP HANA because less data has to be transferred. Q02 is faster when not executing operations in SAP HANA because in this case the additional setup costs are greater than the gain in transfer time.

The optimizer for operations in SAP HANA estimates the number of rows in the result set in the database for the positive decision [EST_DBTRANS_POSITIVE_DECISION] and for the negative decision [EST_DBTRANS_NEGATIVE_DECISION]. The estimation only includes those parts of the query that are executed differently with a positive or negative decision. The estimation takes global filters into account.

The system calculates a reduction factor by which the result set in the database is smaller with a positive decision: [EST_DBTRANS_NEGATIVE_DECISION] / [EST_DBTRANS_POSITIVE_DECISION].

If this reduction factor is greater or equal to the "minimum reduction factor", and [EST_DBTRANS_NEGATIVE_DECISION] is greater or equal to the "minimum DBTRANS (negative decision)", the optimizer accepts for operations in SAP HANA that the benefit of the smaller result set is greater than the additional setup costs for the enhanced expert mode. The optimizer thus makes positive decisions for operations in SAP HANA, meaning that the query is to be executed with the Enhanced Expert mode. Otherwise, the optimizer decides negatively for operations in SAP HANA, and the query is executed with the *Optimized Access (3)* mode.

You can change the "minimum reduction factor" and "minimum DBTRANS (negative decision)" in the Implementation Guide (transaction SPRO).

> **Note**
>
> *SAP Reference IMG* > *SAP BW/4HANA Customizing Implementation Guide* > *SAP BW/4HANA* > *Analysis* > *Settings for Performance of Analytic Manager*

If the "minimum reduction factor" and "Minimum DBTRANS (negative decision)" have values 0 or 1, the optimizer is deactivated for operations in SAP HANA The queries are then always executed with enhanced expert modes. You can use the **Learning Program for the Optimization of Operations in SAP HANA** to generate proposals for the "minimum reduction factor" and "minimum DBTRANS (negative decision)".

The statistics event *Effective TREXOPS* specifies the current value of the query property *Operations in SAP HANA* (used for executing the navigation step) in the column *Event Prop*. If the optimizer is used, this is the result of the optimizer, provided that this value could be applied to the navigation step.

> ### Example
>
> The optimizer concludes that the query is to be executed with *Formulas Calculated in SAP HANA with Complex Currency or Unit of Measure (8)* If the navigation step to be executed does not contain a formula however, it is executed with the maximum possible value, in this example *Expert: Exception Aggregation (Standard) (6)*:

Related Information

Operations in SAP HANA [page 419]
Learning Program for the Optimization of Operations in SAP HANA [page 422]
Configuring the Learning Program [page 423]
Evaluation of Learning Results [page 424]

4.10.2.16.2.7.1.1 Learning Program for the Optimization of Operations in SAP HANA

The learning program (transaction code RSR_LT0) determines the optimum reduction factor that you can use as the default value in the Implementation Guide in the performance settings for the analytic manager.

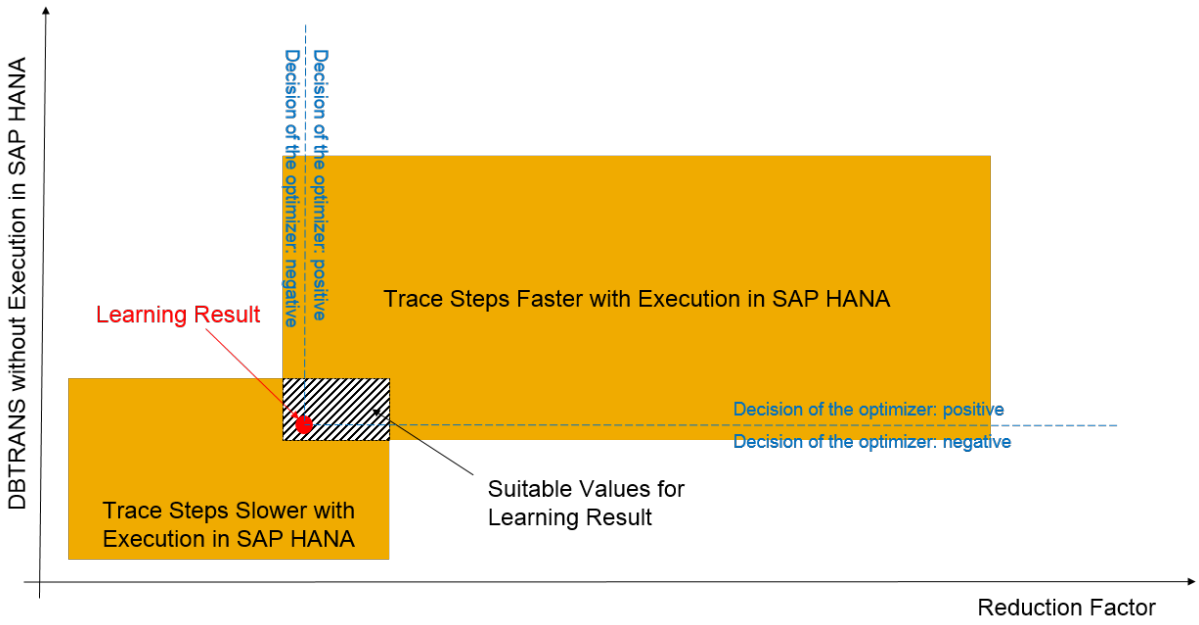
The learning program executes a series of trace steps.

The trace steps are divided into those that are quicker with execution of operations in SAP HANA, and those that take longer. Trace steps that do not contain any operations in SAP HANAor only have a non-relevant runtime difference are not included. SAP HANA

For each trace step the result set of the database (DBTRANS) is included without execution of operations in SAP HANA and the reduction factor of the result set of the database with / without execution of operations in SAP HANA

The optimum "minimum reduction factor" and the optimal "minimum DBTRANS" (negative decision) are in the area in which both trace step sets intersect. The learning program then determines the appropriate values within this range, so that at least 90% of the cases with operation in SAP HANAare faster and performance is increased.

The following figure illustrates this:



Related Information

Configuring the Learning Program [page 423]
Evaluation of Learning Results [page 424]
Optimizer for Operations in SAP HANA [page 421]

4.10.2.16.2.7.1.1.1 Configuring the Learning Program

You can specify the key figures with which the learning program is to execute the trace steps.

Context

You are on the *Measure* tab.

Procedure

1. Enter a test package that you created in the trace tool environment in the CATT trace area. Input help is provided.

2. Enter a minimum runtime difference. You can do this in seconds or as a percentage.

   If a trace step requires the same runtime with and without executing operations in SAP HANA, it is not worth including it in the decision.

   Specifying a tolerated difference is useful to rule out measurement errors. For example, if you enter 0.5 (seconds), it means that only those trace steps that have a runtime greater than or equal to 0.5 (seconds) are included.
3. Enter the number of parallel processes for the execution of the trace.

   The recommendation is the value 1, since a higher value might serialize processes in the database and this could lead to incorrect learning results.
4. Decide whether the query before the calculation is to be executed once, which means that the caches in the database are filled once and the query is generated once.
5. Execute the job.

Next Steps

Choose the *Analyze Jobs* tab:

Related Information

Evaluation of Learning Results [page 424]

4.10.2.16.2.7.1.1.2 Evaluation of Learning Results

On the tab page *Analyze Jobs* you will find the results of the learning program.

The table gives an overview of all jobs in the selected period. If a job has finished (see the *Status*column), you can display the learning results by choosing *Analyze*.

Screen area *Learning Results* for the Optimizer for Operations in SAP HANA

- Default value for the "minimal reduction factor"
- Default value for "Minimum DBTRANS (negative decision)"

> ⓘ Note
>
> If you want to use these default values in the Implementation Guide, choose *Save*.

Screen area *Operations in* SAP HANA

- Main measurement (e.g. *Offensive (P)*
- Reference measurement (e.g. *Expert: Optimized Access (3)*:

Table *Trace ID*

The adjacent table shows the traces with different measurement that have been made (**green**). For these traces, you can display the relevant query runtime statistics.

Traces that have a **yellow** status do not contain any operations to be executed in SAP HANA and are not relevant for this learning program.

Traces in which an error occurred are marked **red**. You can execute these again by choosing *execute*.

Table *Trace Steps Executed in Different Ways:*

In the lower screen area, you can see the analysis of the trace steps that contain operations to be executed in SAP HANA and were executed differently in the two measurements. You can see whether the traces were faster with or without execution of operations in SAP HANA, which decision the optimizer made, and whether this decision was the correct one. In the following columns you can find the technical key figures for the optimization.

Related Information

Configuring the Learning Program [page 423]
Optimizer for Operations in SAP HANA [page 421]

4.10.2.16.2.7.2 Exception Aggregation in SAP HANA

Supported Aggregations

The following exception aggregations can be calculated in SAP HANA:

Supported Aggregations

Exception Aggregation	SAP HANA
Count All Values [CNT]	X

Exception Aggregation	SAP HANA
Constant with exception aggregation SUM	X
Multiple-dimensional exception aggregation SUM for a constant	X
Total [SUM]	X
Minimum [MIN]	X
Maximum [MAX]	X
Count All Values <> 0 [CN0]	X
Average of All Values [AVG]	X
Average of All Values < > 0 [AV0]	X
First Value [FIR]	2156123
Last Value [LAST]	2156123
No aggregation if more than one record arises [NO1]	SAP HANA 1.0 SP11 (or higher)
No aggregation if more than one value arises [NO2]	SAP HANA 1.0 SP11 (or higher)
Standard Deviation [STD]	SAP HANA 1.0 SP12 (or higher)
Variance [VAR]	SAP HANA 1.0 SP12 (or higher)

Non-Supported Functions for Individual Key Figures

Calculating exception aggregation in SAP HANAis **not** possible if the key figure uses one of the following functions:

- Non-cumulative key figures
- Key figures with elimination of internal business volume
- Virtual key figures
- Stock coverage key figures
- If the key figure uses a currency translation or unit conversion that uses an InfoObject to deterrmine the target currency/unit.

Commutative Formulas before Aggregation

Commutative formulas are formulas that can be calculated before or after aggregation and always have the same result.

> ### Example
>
> a * 5: If key figure a has aggregation total [SUM], this formula is a commutative formula.
>
> a * b: With multiplication of the key figures a and b**before** aggregation, this produces a different result. This is therefore **not** a commutative formula.

If a commutative formula has a supported exception aggregation, and commutative formulas are calculated **before** aggregation in accordance with the query properties, this formula is calculated in SAP HANA.

Non-Supported Functions in a Query

Calculating exception aggregations for a query in SAP HANAis **not** possible if the query uses one of the following functions:

- Determination of time-dependent hierarchies for an event (temporal hierarchy join)
- Virtual characteristics
- Artificial characteristic "currency/unitt" (1CUDIM)

Pushdown of additional features

If it is possible to calculate an exception aggregation in SAP HANA, the following calculations are performed in SAP HANA even if they are not in an exception aggregation:

- Currency translations with a variable time reference from a special InfoObject are calculated in SAP HANA.
- If the source unit of quantity is determined from the data record,
  - the system uses a reference InfoObject in the SAP HANA to calculate unit conversions.
  - As of SAP HANA 2.0, the system uses a reference InfoObject and central units of key figures (T006) in the SAP HANA to calculate unit conversions.
- Variables with replacement from the attribute value of a characteristic

Variables with replacement from the attribute value of a characteristic

Variables with processing type Replacement path, with replacement from the attribute value of a characteristic are supported as of SAP HANA1.0 SP11 (or higher). The following prerequisites must be met:

- The replacement does **not** take place after aggregation.
- The attribute is either a key figure or a characteristic. If the attribute is a characteristic, it must be either a currency or a unit, or it must have one of the following data types:
  - CHAR - Character String
  - NUMC - Numerical Text
  - DATS - Date in Format YYYYMMDD
  - TIMS - Time in Format HHMMSS
  - SNUMC - Signed numeric text (is supported up to a length of 20 letters as of SAP HANA 1.0 SP12)

> **Note**
>
> In the case of compound characteristics, note that this also applies to the characteristic that serves as the compounding parent.

- In the case of the characteristic or - in the case of referencing characteristics - the referenced characteristic, one of the following conditions applies:
  - It has no master data read class.
  - If it does have a master data read class, this allows the use of master data tables. These include master data read classes of the time characteristics delivered by SAP, provided that the time characteristic has at least one navigation attribute. In this case, the system does **not** support transitive attributes.
  - For all other characteristics, the system supports the replacement of the artificial attributes “Key” and “Characteristic Reference (Constant 1)” only.

4.10.2.16.2.7.3 Formulas Calculated in SAP HANA

Supported Aggregations

The following exception aggregations are supported as of SAP HANA 1.0 SP11:

Supported Aggregations

Exception Aggregation	SAP HANA
Count All Values [CNT]	X
Total [SUM]	X
Minimum [MIN]	X
Maximum [MAX]	X
Count All Values <> 0 [CN0]	X
Average of All Values [AVG]	X
Average of All Values < > 0 [AV0]	X
First Value [FIR]	2156123
Last Value [LAST]	2156123
No aggregation if more than one record arises [NO1]	X
No aggregation if more than one value arises [NO2]	X
Standard Deviation [STD]	As of SAP HANA 1.0 SP12
Variance [VAR]	As of SAP HANA 1.0 SP12

Supported formula operators

Calculating formula exception aggregations in SAP HANA is possible for the following operators:

Supported formula operators

Operator	NODIM-like	Requires NODIM	Floating comma not supported
Plus [+]			
Minus [-]			
Divide [/]			
Multiply [*]			
Value without units or currencies [NODIM]	X		
Whole division [DIV]			
CURRENT MEMBER result [CMR]			
Counter [COUNT]	X		
Delta [DELTA]	X		
Absolute value [ABS]			
Maximum (MAX)			
Minimum (MIN)			
+/- sign of the operator [SIGN]	X		
Whole number value of the operand [TRUNC]			
Less than [<]	X		
Less than or equal to [<=]	X		
Not equal to [<>]	X		
Equal to [==]	X		
Greater than [>]	X		
Less than or equal to [<=]	X		
Logical AND [AND]	X		

Operator	NODIM-like	Requires NODIM	Floating comma not supported
If condition [IF]	1		
Logical NOT [NOT]	X		
Logical OR [OR]	X		
Logical exclusive OR [XOR]	X		
Smallest whole number value that is greater than operand [CEIL]			X
Greatest whole number value that is less than operand [FLOOR]			X
Decimal part of operand [FRAC]		X	X
Remainder with whole-number division [MOD]	2		
High (**)		X	
Potential function [EXP]		X	
Logarithm [LOG]		X	
Logarithm for basis 10 [LOG10]		X	
Root [SQRT]		X	
NDIV0(x): Is equal to 0 on division by 0, else x [NDIV0]		X	
NOERR (x): Is equal to 0 with undefined calculations, else x [NOERR]		X	
Maximum of 0 and operand [MAX0]		X	
Minimum of 0 and operand [MIN0]		X	

Explanation of the table columns:

- **NODIM-like**:

    - X: The result of the operator never has a currency or a unit, even if the operands have a currency or unit. This means that the operator **ignores** the currencies or units of all operands.
- <n>: The curernncy or unit of the nth operand is **ignored**. n = 1 for example therefore means that the currency or unit of the first operand is **ignored**.
- **Requires NODIM**: Operator O1 is only supported if its operands do not have currency or unit, or the operation of O1 is an operand of a NODIM-type operator O2, for which a currency or unit is ignored.
- **NODIM type**: The operator is only supported if no operands are of type floating comma.

Currencies and Units

A formula with exception aggregation can only be calculated in SAP HANA if the formula result does not have a currency or a unit, or the currency or unit of the formula result is taken over from another individual operand.

> Example
>
> `a * b`: If `a` contains a currency or unit, and `b` does not, this formula can be calculated in SAP HANA.
>
> `a * ( b = 0 )`: Because the operator = operator is NODIM-like, the formula can be calculated in SAP HANA.
>
> `a + b`: This formula cannot be calculated in SAP HANA if `a` or `b` has a currency or unit.

4.10.2.16.2.7.4 Formulas Calculated in SAP HANAwith Complex Currency or Unit of Measure

The system supports the calculation of formulas with complex currency or unit of measure in SAP HANA

Supported Aggregations

The system supports the same exception aggregations as in the *Formulas Calculated in SAP HANA (7)* mode.

Supported Formula Operators

The system supports the same operators as in the *Formulas Calculated in SAP HANA (7)* mode.

You also require the following formula operators **not** NODIM:

- *NDIV0(x): Is equal to 0 on division by 0, else x [NDIV0]*
- *NOERR(x): Is equal to 0 with undefined calculations, else x [NOERR]*
- *Maximum of 0 and operand [MAX0]*

- *Minimum of 0 and operand [MIN0]*

Currencies and Units

A formula with exception aggregation can only be calculated in SAP HANA if the result of the formula and all subtotals have up to five currencies and units of measure. The currencies and units of measure can have an exponent.

> ### Example
>
> `a * b`: The factors a and b have one currency or unit of measure. The result of the formula has two currencies or units of measure. The formula can therefore be calculated in the SAP HANA.
>
> `a / b`: The operands have one currency or unit of measure. The result of the formula has two currencies or units of measure:
>
> - Currency or unit of measure of operand a (with exponent 1)
> - Currency or unit of measure of operand b (with exponent -1)
>
> The formula can be calculated in the SAP HANA.
>
> `a + b`: The summands a and b have one currency or unit of measure. The result of the formula has one currency or unit of measure. The formula can therefore be calculated in the SAP HANA.
>
> - If a and b have the same currency or unit of measure, this is the currency or unit of measure of the result of the formula.
> - If a und b have different currencies or units of measure, the currency or unit of measure of the result of the formula is mixed.
>
> `a * b/c * d * e/f`: The operands `a, b, c, d, e, f` have one currency or unit of measure. If all of these currencies and units of measure are different, the result of the formula has six currencies and units of measure. This means that this formula has more than five currencies and units of measure and that it therefore **cannot** be calculated in the SAP HANA.

Different result in case of mixed currencies and units of measure

The system handles currencies and units of measure separately if a formula **cannot** be calculated in the SAP HANA. Calculation of formulas in the SAP HANA makes no distinction between currencies and units of measure however.

For results of formulas with currencies and units of measure, this produces different results if currencies and units of measure are mixed.

> ### Example
>
> 10 EUR * (2 m + 4 s): The result of the formula has one currency (EUR) but a mixed unit (m + s = *),
>
> - Result of the formula without calculation of the formula in the SAP HANA: 60 EUR *
> - Result of the formula with calculation of the formula in the SAP HANA: 60 *

(10 EUR - 5 USD) * (2 m + 4 s): The result of the formula has a mixed currency or unit of measure.

- Result of the formula without calculation of the formula in the SAP HANA: 30 EUR **
- Result of the formula with calculation of the formula in the SAP HANA: 30 *

4.10.2.16.2.7.5 Conditions Calculated in the SAP HANA

The system supports the calculation of conditions in the SAP HANA.

Application Scenario

A telecommunications company would like to gather information about the top three of their 20 million customers. The following section shows how can be down both with and without calculation on the SAP HANA:

Procedure without calculation of the condition on the SAP HANA

1. The system retrieves data for all 20 million customers from the database.
2. The system calculates exception aggregations, conversions, formulas and so on for all 20 million customers.
3. The system determines the top three customers.
4. The system displays the data for the top three customers only.

Disadvantage: The application server retrieves the data for all 20 million customers and performs calculations with this data, but the does **not** see the data for the remaining 19,999,997 customers in the query result.

Procedure with calculation of the condition on the SAP HANA

1. SAP HANA determines the top three customers.
2. The system retrieves data for the top three customers from the database.
3. The system calculates exception aggregations, conversions, formulas and so on for the top three customers.
4. The system displays the data for the top three customers.

Advantage: To begin with, the TopN condition is calculated on the SAP HANA. The application server then retrieves the data for the top three customers only and performs calculations with this data, which the customer can then see in the query result.

Features

Every condition is applied at the corresponding subtotal level. This evaluation process runs from the coarsest subtotal level through to the finest one on every axis. If conditions can be calculated in the SAP HANA, the system runs a pre-query for every corresponding subtotal level. The pre-query returns the corresponding characteristic tuple. The results of the pre-query are later used as a filter.

As coarser subtotals also contain data that lies outside the condition filter of a finer subtotal level, the data from the finer subtotal level cannot be aggregated to the coarser subtotal levels. The data for the coarser subtotal levels needs to be retrieved separately from the database instead.

Prerequisites

For the calculation of conditions in SAP HANA, the following prerequisites apply:

- It must be possible to calculate all condition parameters in SAP HANA.
- NULL values never meet the condition. This must be ensured by one of the following cases:
  - NULL values that are considered to be "0" when conditions are processed cannot meet a condition.
  - If the *Ignore NULL Values* flag has been selected in the condition definition, NULL values cannot meet a condition either, as they are ignored (see Tab: General [page 368]).

> **Note**
>
> You can either set the *Ignore NULL Values* flag for individual conditions of a query using the BW Modeling tools, or can use report RSO_RES_QD_CONDMODE to set multiple queries for all conditions.

Supported Operators

A condition parameter can only be calculated in SAP HANA if it uses a supported parameter operator.

Supported ranking operators

- TopN
- BottomN
- TopSUM
- BottomSUM

Supported having operators

- Equal
- NotEqualTo
- LessThan
- GreaterThan
- LessEqual
- GreaterEqual
- Between
- NotBetween

Structure member that is used by a condition parameter

A condition parameter can only be calculated in SAP HANA if the structure member used can be calculated in SAP HANA.

- The structure member can be calculated in SAP HANA with the option *Formulas calculated in SAP HANA with complex currency/unit*.
- The structure member fulfills all prerequisites for the calculation in SAP HANA with the option *Formulas calculated in SAP HANA with complex currency/unit* except the prerequisite that it has an exception aggregation.

Condition parameters using formulas

A condition parameter that uses a formula as a structure member can only be calculated in SAP HANA if the following prerequisites are met:

- All operands in the formula have the same context of a constant selection.
- All operands of the formula have the same current member context.
  A current member context is defined by the current member variable, the selected range, the offset, and the characteristics that are used for the exception aggregation.
- The formula result is NULL if all formula operands are NULL.

Constant Selection and Current Member Context of Condition Parameters

A condition can only be calculated in SAP HANA if all condition parameters have the same unique constant selection and current member context.

Limitations for conditions using ranking operators

If more than one condition with ranking operators is applied to the subtotal level, **no** conditions can be calculated for this subtotal level in SAP HANA.

If only one of the conditions applied to a subtotal level uses ranking operators, only this condition will be calculated for this subtotal level in SAP HANA.

If one of the conditions with a ranking operator is applied to a structure element that uses a current member interval, this condition **cannot** be calculated in the SAP HANA.

Performance

If the conditions significantly reduce the query result set, the calculation of conditions in SAP HANA results in improved performance.

Improved performance is unlikely for example if a TopN = 3 condition is calculated in SAP HANA, and there are only five characteristic values.

4.10.2.16.2.7.5.1 Different Condition Result with Exception Aggregation Counters (CNT or CN0)

This section describes a scenario that illustrates how the Analytic Engine handles the exception aggregations **Counter (all values) [CNT]** and **Counter (values unequal to 0) [CN0]**.

Scenario

A query contains a key figure or a formula K1 with exception aggregation Counter (all values) [CNT] or Counter (values unequal to 0) [CN0].

For a certain characteristic value combination, no value is booked for K1. The characteristic value combination therefore only shown in the query result because there is a posting for another structure element or because unposted values are displayed in the query result.

The Analytic Engine writes an artificial value 0 to the query result for K1 or uses this value as an operand in a formula F2 that does not have exception aggregation.

A condition exists for K1 or F2.

Value 0 meets the condition.

The condition result is different if conditions like these are calculated in the SAP HANA because the artificial 0 values introduced by the Analytic Engine are **not** known during processing of a condition in the SAP HANA.

In the case of a threshold condition, the condition result can be smaller because characteristic value combinations with an artificial 0 are lost in the condition result.

In the case of a ranked list condition, the condition result can be different because only characteristic value combinations with a real value - and not characteristic value combinations with an artificial 0 - are part of the condition result.

Scenario Example

Let us assume that a query has the following result:

Year	Quantity	Quantity in Germany	Quantity in Germany, CNT on Product
2015	100	50	2
2016	200	20	1
2017	150		0

Now let's apply the condition **Amount in Germany, CNT on Product** <= 1

If the conditions are **not** calculated in the SAP HANA, the query result is as follows:

Year	Quantity	Quantity in Germany	Quantity in Germany, CNT on Product
2016	200	20	1
2017	150		0

If the conditions are calculated in the SAP HANA, the query result is different because the artificial 0 from 2017 is ignored during processing of the condition in the SAP HANA:

Year	Quantity	Quantity in Germany	Quantity in Germany, CNT on Product
2016	200	20	1

If structure element **Quantity** is not selected in the query, this difference does not apply. In this case, the results are the same, regardless of whether or not the condition was processed in the SAP HANA:

Year	Quantity in Germany	Quantity in Germany, CNT on Product
2016	20	1

Recommendation

In order to avoid artificial 0 values and to achieve the same condition result regardless of whether or not the condition was processed in the SAP HANA, you can proceed as follows:

Instead of **Counter (all values) [CNT]**, use exception aggregation **Summation [SUM] on the constant 1** with the same filter as though it were the operand of exception aggregation **Counter (all values) [CNT]**.

Instead of **Counter (values unequal to 0) [CN0]**, use exception aggregation **Summation [SUM]** of a formula that uses the operator **Counter [COUNT]** and the operand of exception aggregation **Counter (values unequal to 0) [CN0]** as the operand.

Recommendation Example

Let us assume that a query has the following result:

Year	Quantity	Quantity in Germany	Quantity in Germany, SUM on Product
2015	100	50	2
2016	200	20	1
2017	150		

Now let's apply the condition **1 in Germany, SUM on Product** <= 1

The query results are the same, regardless of whether or not the condition was processed in the SAP HANA:

Year	Quantity	Quantity in Germany	1 in Germany, SUM on Product
2016	200	20	1

4.10.2.16.2.8 Materializing the Intermediate Result of the Query

You can change the standard settings for processing the intermediate results for existing queries.

Prerequisites

This property is only relevant if query property *Operations in SAP HANA* is set to *Expert: Exception Aggregation (Standard) (6)* or higher.

Context

If query property *Operations in SAP HANA* is set to *Expert: Exception Aggregation (Standard) (6)* or higher, a significant part of query processing takes place in the SAP HANA database.

The intermediate results of the processing are calculated in this processing. If the query contains many characteristics with intermediate results in the drilldown, these intermediate results can be used again during the result calculations.

It is generally quicker to recalculate the intermediate results each time, since optimizations can then be used at the level of the intermediate results. In some cases, however, it is quicker to materialized the results sets of the intermediate results.

These exception cases occur when more than one of the following properties apply:

- A large number of characteristics in the drilldown, each characteristic with intermediate results.
- A great reduction in data due to the standard aggregation.
- Very little reduction in data due to filters.

Procedure

1. Deselect the *InfoProvider Setting* option.
2. You can activate or deactivate the *Materialize Intermediate Results of the Query* options specifically for the selected query.

Example

The InfoProvider contains two billion rows. The data includes the following:

- 500 unique material numbers
- 20 countries
- Five billing types
- 12 months

The query counts the material numbers per country, per country and billing type, and per country, billing type, and month. The set of intermediate results has the most finely grained tuples required in this query. If all combinations of characteristics exist, the size of the intermediate result is 500*20*5*12 = 600000 rows.

This intermediate results contains over 1000 times fewer records than the InfoProvider. This means that it probably cheaper to materialize and reuse the intermediate result than use the optimizations that can be applied when calculating the individual levels of the intermediate results from the InfoProvider data. In this case, therefore, the recommended action is to materialize the intermediate result.

Related Information

Operations in SAP HANA [page 419]

4.10.2.16.2.9 Using the Selection of Structure Elements

You can use the query property *Use Selection of Structure Elements* to ensure that data is read only for exactly the structure elements that are being read. This means that the drilldown characteristics can display fewer attributes.

Context

The *Use Selection of Structure Elements* option influences system performance. This function should therefore normally be activated.

Example

Example of a Simple Query Without Selection

Country	Sales Volume (DE)	Sales Volume (US)
DE	50	
US		60

Query with Selection on Structure Element "Sales Volume (DE)" Without Selection of Structure Elements

Country	Sales Volume (DE)
DE	50
US	

Query with Selection on Structure Element "Sales Volume (DE)" and Selection of Structure Elements

Country	Sales Volume (DE)
DE	50

Technically speaking, the system only passes to the database the selections and key figures of the columns (or structure elements, to be more precise) that are required to determine the filtered elements. This means that cells that only exist because of non-selected structure elements are not displayed.

Structure elements with constant selection are an exception, as they do not create any objects (see). If the selection contains a structure element with constant selection, data for further structure elements outside the selection are requested.

Example

Example of a Simple Query Without Selection

Country	Sales Volume (Constant Country Selection)	Sales Volume (DE)	Sales Volume (US)
DE	300	50	
US	300		60

Query with Selection on Structure Element "Sales Volume (Constant Country Selection)" and "Sales Volume (DE)" and Selection of Structure Elements

Country	Sales Volume (Constant Country Selection)	Sales Volume (DE)
DE	300	50
US	300	

If you deactivate this function, the data is read for the entire structure or for both structures from the database. This function therefore can also **not** be activated if the query has read mode 'A' (see Read Mode [page 415]).

You can expect a significant improvement in performance from using this function in the following scenarios:

- You are using the SAP HANA database or the SAP BW Accelerator.
- You are using queries that contain a large number of selections or restricted key figures, which are not all displayed in the initial drilldown.
- You execute a lot of operations where you filter using selections or restricted key figures, such as the "Filter and Drill Down By" function.

Deactivating this function does not normally bring any advantages.

Related Information

Reading Child Members with Hierarchy Structure Nodes [page 441]

4.10.2.16.2.10 Reading Child Members with Hierarchy Structure Nodes

If you have activated the *Use Selection of Structure Elements* setting, you can use this setting to specify which data-driven cells are displayed if the query has a hierarchy on a structure.

Use

The following example illustrates this:

Example

Month	January	February	March
Amount	EUR 10	EUR 20	EUR 30

Month	January	February	March
Amount in January	EUR 10		
+- Amount in February		EUR 20	
+- Amount in March			EUR 30

Amount in Feb is a child of **Amount in Jan**, and **Amount in March** is a child of **Amount in Feb**.

If **Amount in Jan** is filtered, the result set will only consist of the January column. If the **Amount in Jan** node is expanded now, the result set must consist of the January and February columns for **Amount in Jan** and **Amount in Feb**.

Most clients however do not recognize a situation where a drilldown on one axis can result in more data on another axis. This drilldown therefore normally only shows the January column.

If the *Also Read Child Members* setting is activated, a filter on a structural element will implicitly include all of its hierarchical successors in the filter. In the example above, a filter on **Amount in Jan** would return the columns for Januar, February and March, as **Amount in Feb** and **Amount in March** are predecessors of **Amount in Jan**.

Related Information

Using the Selection of Structure Elements [page 440]

4.10.2.16.2.11 Optimization Mode

With the *Optimization Mode* property, you can define how often the OLAP processor memory requirement is optimized for existing queries.

Context

If you do not make any other settings, the OLAP processor specifies the optimal memory requirements for the data processed in the OLAP processor. This is done after the query is executed for the first time and then again 31 days after the last optimization. A report is generated automatically.

The following modes exist:

- **Query Will Be Optimized after Generation (0)**: Default setting.
- **Query Optimization with Individual Periods in Days (1)**: You can define the **optimization period** as required. You can use this setting if the InfoProvider data is periodically fully reloaded or rebuilt.
- **Query Optimization Inactive (9)** : This setting is usually only used for test and support purposes or in exceptional cases.

> **Note**
>
> For more information, see SAP Note 1137994.

Related Information

Optimization Periods [page 443]

4.10.2.16.2.12 Optimization Periods

If you have set the optimization mode to *Query Optimization with Individual Period in Days (1)*, you can set any optimization period of your choosing.

You can use this setting if the InfoProvider data is periodically fully reloaded or rebuilt.

Related Information

Optimization Mode [page 442]

4.10.2.16.2.13 Statistic Details Display

The system displays the detail level for statistic details of query runtime statistics as defined in the statistic properties maintenance transaction (RSDDSTAT) for object type Query.

Related Information

You can find the relevant information under *Statistic Detail Level for Object Type Query* in the documentation for maintenance of statistic properties.

4.10.2.16.2.14 Document Support

You can use query property *Document Support* to specify whether documents can be created for the query.

If you set value **Yes** for query property *Document Support*, you can use the *Default Document Application ID* field to stipulate which document store the documents will be stored in. You can select from all document stores to which the InfoProvider that the query is based on is assigned. This covers the following InfoProviders:

- DataStore Object
- CompositeProvider

If you do not select a document store, and keep the value **[ ]** instead, the user has to assign a document store at runtime.

Related Information

Document in the Document Store [page 110]
Creating a Document Store [page 314]

4.10.3 Opening and Changing Existing Queries

Context

There are various ways of opening a query:

- From your favorites (provided that you have added the query to your favorites)
- Using *Open BW Object*
- Using *BW Object Search*. Here, you can refine the search to type *ELEM [Query Element]* under *Advanced Filter*.
- From the *BW Repository* in the Project Explorer view

If you have changed a query definition, an asterisk is displayed in front of the technical query name. This asterisk remains until you save your changes.

Related Information

Working with Queries [page 316]
Searching for and Opening Objects with the Open BW Object Dialog Box [page 24]
Searching for SAP BW/4HANA Objects [page 25]
Saving Queries [page 374]

4.11 Creating a Link Component

You want to create a target component (for example target query Q_B) that is kept synchronous when changes are made to the corresponding source component (for example source query Q_A).

Procedure

1. Open the source component.

   In our example, open source query Q_A.
2. Press the *Show Linked Components View* button.

   The system displays the *Linked Components* tab.
3. Open the context menu for the source component and choose *New Component*.

   The *New Component* dialog box appears.
4. Enter the name of the target component and a description.
5. Specify the name of the target InfoProvider. Alternatively, you can press CTRL + Space to call input help and select the required InfoProvider.
6. Choose *Create Proposal*.

   The system displays the mapping proposal in the *Mapping Details* table: In the *Source* column, you see the definition of the source component. In the *Target* column, you see the matching elements in the target component as determined by the system. If the system does not find any corresponding fields, it indicates this in the status information.
7. If you want to change a mapping, click in the field of the target element.

   A dialog box appears where you can select a different element in the target InfoProvider to map.
8. To have your changes checked by the system, choose *Validate*.

   The system performs type and existence checks.
9. If all statuses are displayed as green, choose *Create*.

   You are returned to the *Linked Components* tab. The system displays the target component you have created as hierarchically assigned to the source component.

Results

Any changes you make to the source component now will be kept synchronous in the target components by the system.

Related Information

Linked Component [page 140]
This link takes to you to the documentation for query concepts.
Apply Changes from the Source Component [page 446]
Changing a Target Component [page 447]
These links take you to the documentation for further tasks.

4.11.1 Apply Changes from the Source Component

You have created a linked target component and made changes to the source component. The main advantage of having a linked component is that it allows you to automatically apply changes made in the source component.

Procedure

1. Open the source component.

   On the *Linked Components* tab, the target component icon is modified, thus indicating that the target component is currently not synchronous with the source component.
2. To start synchronization, choose *Synchronize* from the context menu for the source component.

   The system indicates the progress of the synchronization in a separate dialog.
3. To check that synchronization has been performed successfully, open the target component and choose *Refresh*.

Related Information

Creating a Link Component [page 445]

4.11.2 Changing a Target Component

You have created a linked target component and would now like to change it.

Context

As linking two components only makes sense if the source component and target component correspond with one another, changes to the target component are only possible if these meet the conditions for linking.

Procedure

1. Open the context menu for the link target component on the *Linked Components* tab and choose *Open*.

   The maintenance screen for the target component appears.
2. Under *General*, press the *Edit mapping and synchronize with source component* button.

   The Mapping dialog box appears.
3. Make the required changes.

   You can change the description or the mapping for example.
4. To have your changes checked by the system, choose *Validate*.

   The system performs type and existence checks.
5. If all statuses are displayed as green, choose *Synchronize*.

   You are returned to the *Linked Components* tab.

Results

You have changed the target component and synchronized the link connection.

If you want to make further changes to the target component, you have to decouple the link connection. To do this, press the *Decouple* button under *General*. The target component will then no longer be connected with the source component. You can now make any changes as required. The previous target component is completely independent of the source component.

Related Information

Creating a Link Component [page 445]

4.12 Using the Open Hub Destination to Distribute Data from the SAP BW/4HANA System

The open hub destination allows you to distribute data from a SAP BW/4HANA system to non-SAP data marts, analytical applications, and other applications. It allows you to ensure controlled distribution across several systems.

Prerequisites

You have already loaded data into an InfoProvider or DataSource.

Context

The open hub destination defines the target that the data is relayed to. Any objects supported by the data transfer process can be used as open hub data sources.

Database tables and flat files can serve as open hub destinations. The database table can be located in SAP BW/4HANA or in another database system that is supported by SAP NetWeaver. The data can be extracted from the database to a downstream system.

Procedure

1. **Creating Open Hub Destinations**

   Create an open hub destination with the required target.
2. **Creating Transformations**

   Create a transformation for your open hub destination. Note that not all rule types are available in the transformation for an open hub destination. Master data reading, time conversion, currency translation and unit conversion are not available. If certain prerequisites are met, execution in SAP HANA is supported.
3. **Creating a Data Transfer Process**

   Create a data transfer process for your open hub destination.

4.12.1 Creating Open Hub Destinations

The open hub destination defines the target that the data is relayed to.

Procedure

1. You are in the BW Modeling tools. In the context menu for your BW project, choose *New* > *Open Hub Destination*.

   You can also create open hub destinations from the data flow. To do this, go the *BW Object Types* in the editing screen for your data flow, select the type *Open Hub Destination* and drag it to *Details*. Then open the context menu for this open hub destination and choose *Create Persistent BW Object*.

   Whenever you create an open hub destination, you are guided through the process by a wizard. More information: Creating an Open Hub Destination Using the Wizard [page 449]
2. On the *General* tab, you can change the open hub destination type again if required. The other settings that you can make on this tab depend on this.
   - Selecting Database Tables as Destinations [page 450]
   - Selecting Files as Destinations [page 453]
3. Edit the field list on the *Details* tab. More information: Edit Fields [page 454]
4. Activate the open hub destination.

4.12.1.1 Creating an Open Hub Destination Using the Wizard

Whenever you create an open hub destination, you are guided through the process by a wizard.

Context

Procedure

1. 1. You are in the BW Modeling tools. In the context menu for your BW project, choose *New* > *Open Hub Destination*.

   You can also create open hub destinations from the data flow. To do this, go the *BW Object Types* in the editing screen for your data flow, select the type *Open Hub Destination* and drag it to *Details*. Then open the context menu for this open hub destination and choose *Create Persistent BW Object*.

2. Select an InfoArea. To ensure that you can find your open hub destination again without difficulty, we recommend setting the *Add to Favorites* flag.
3. Enter a technical name and a description. We recommend using the object that you want to update data to the open hub destination from as the template.
4. Select the required open hub destination type.
5. When you have finished making your settings, choose *Finish*. The editor appears.

4.12.2 Selecting Database Tables as Destinations

You can write data from the BW system to a database table via the open hub destination.

Context

You can also connect another database. The data is then also written to the generated table in this database. This allows you to publish data directly from the BW system to another system. There is no need to replicate the data, and no third-party tools are required. If you are using SAP HANA, you can also process the data subsequently using SAP HANA functions.

The following databases are supported: SAP HANA, SAP IQ (Intelligent Query), SAP ASE (Adaptive Server Enterprise), Oracle, IBM DB2, DB4, DB6, Microsoft SQL Server, Sybase.

Procedure

1. Choose *Database Table* as the type of destination. When you activate the open hub destination, the system generates a database table. The generated database table has the prefix */BIC/OHxxx*, xxx being the technical name of the destination.
2. By selecting the *Send Notification to RFC Destination* flag, you can have a notification sent after successful execution of the DTP (fill or destination). When this happens, module RSB_API_OHS_ETL_NOTIFY is called by RFC. This module must therefore also be implemented in the target destination with exactly the same name and interface. Once this has been confirmed, the DTP is continued.
3. By selecting the *DTPs Wait for Status Set by External Program* flag, you can set the status for a request in the target system to *red* or *green* and append a message. After successful execution, the DTP is stopped in the technical state, meaning that the DTP is executed and fills the destination. The status of the DTP then remains *green* technically and *yellow* overall until module RSB_API_OHS_ETL_SETSTATUS is called. Here you can set the status of the request to *red* or *green* and append a message. The message appears in the DTP log.
4. Under *Connection Type*, you can have the following options:

Connection Type	Description
No Connection (local database)	You create the table in the local database.

Connection Type	Description
Database Connection (DBCON)	You create the table using a connection in a remote database. Under *Database Connection*, enter a previously created database connection so that the data is also written to this database. You can create or edit the connection to the database in the DBA cockpit. For more information about the DBA cockpit, go to *http://help.sap.com/netweaver* > *SAP NetWeaver Platform* > *Application Help* > *Function-Oriented View* > *Database Administration* > *<Your Database>*. In the target system (in the database), a table is created with prefix /BIC/. If the database instance also has an SAP instance, like an SAP ERP system on a SAP HANA database for example, the DDIC object also has to be generated manually. The namespace is also registered when this is done. The DB table name in the remote system is identical to the DDIC name in the local system, or corresponds to the name that would have been created locally on the database. **Note** You have to make sure here that there is no table with the same name in the target system, as this would result in an error when creating the table. The SAP_BASIS version of both SAP systems must also be either identical or compatible, and the codepage (ASCII, Unicode) should match.
Smart Data Access (SDA)	You create the table using Smart Data Access (SDA). This allows you to create a virtual table in the local database that references a remote table in a remote database. The data is processed in the SAP HANA runtime. **Caution** This *connection type* is available for SAP HANA, SAP IQ (intelligent query), and SAP ASE (Adaptive Server Enterprise). No other databases are supported. To be able to use the connections, you - or a member of your IT team - need to edit table RSBOHCONNECTIONS in the SAP GUI via the maintenance view (transaction SM30) by specifying a name for the connection, the remote source, the remote database, and the remote schema. **Note** The database user of the ABAP server requires the two object privileges CREATE VIRTUAL TABLE and REMOTE TABLE ADMIN on the SAP HANA remote

Connection Type	Description
	source (see CREATE VIRTUAL TABLE Statement (Data Definition)).

> → Tip
>
> We recommend choosing *Connection Type***Smart Data Access (SDA)** wherever possible.

5. Under **Extraction Type**, you have the following options:
   - Delete data and insert data records: The fields are overwritten. The table is completely deleted before every extraction and regenerated. We recommend this if you do not want to store the history of the data in the table.
   - Retain data and insert records (only with technical key selected): The data records are inserted. The table is generated just once prior to the first extraction. This allows you to obtain the history of the extracted data.
   - Retain data and change records (only with semantic key selected): You can only do this if you have selected *Semantic Key*. (If you have selected technical key, this option behaves in the same way as "Retain Data and Insert Records".)

> ⓘ Note
>
> Note that the table is always deleted and regenerated if changes are made to the properties of the database table (by adding fields for example).

6. You can choose whether to use a technical key or a semantic key.
   - If you set the *Technical Key* flag, the system generates a unique key from the technical fields (request number, package number, record number). These fields display the individual key fields for the table. These also include the technical name for the request. See Monitoring Data Transfer Process [page 547].

     > ⓘ Note
     >
     > Using a technical key with a target table is particularly useful if you want to extract data to a table that is not deleted before extraction. Where a record is extracted with the same key as a record already in the system, a short dump occurs due to the presence of duplicate records. The technical key for the delta information is also required if the DTP loads deltas.

   - If you set the *Semantical Key* flag, you can explicitly select the key fields. Note however that using a semantic key can result in duplicate records. The records are not aggregated. Instead each extracted record is saved in the table.

4.12.3 Selecting Files as Destinations

You can write data from the BW system to a file via the open hub destination.

Context

Unless you select a different target code page, note that the system writes in UTF-8 by default.

Procedure

1. Choose *File* as the type of destination. The file formats CSV, ASCII, and XML are supported for extraction to files. A control file with information about the metadata is also generated.
2. If you are using Hadoop, you can save the file on the application server as and *HDFS file*. Otherwise you can save it in a local directory. If the file is located on the application server, you can schedule extraction in the background.

   > ⚠ Caution
   >
   > If you save the file locally, the file size must not exceed half a gigabyte. When transferring mass data, you should save the file on the application server.
   >
   > The User Account Control (UAC) in Windows 7 and Windows Vista can create the impression that data is written to a local directory, but no files are actually created. Do not use a directory with restricted authorization.

3. If you want to write the data to an application server from the BW system, you have two options for the file name:
   - File name: The file name is made up of the technical name of the open hub destination and the extension *.CSV*, *.TXT* or *.XML*. You cannot change this name.
   - Logical file name: You can use input help to select a logical file name that you have already defined in Customizing. To do this, you create a logical path and assign a logical file name to it. More information: Defining the Logical Path Name and File Name
     A logical file name can be made up of fixed path information and variables - such as calendar day and time. Logical file names can be transported.
4. If you save the data as an *HDFS file*, you have to specify an RFC connection of type *HTTP* for the Hadoop server that you created previously. The file is saved in the home directory. You can change the name of the file.
5. If you save the file in a local directory, you **cannot** change the name of the file. It is made up of the technical name of the open hub destination and the extension *.CSV*, *.TXT* or *.XML*. The associated control file is given the prefix *S_*.
6. If you select data format XML, you can choose the following formats:
   - SAP Standard XML ist the default SAP XML format. It is optimized for data exchange between SAP and external systems in both directions.

- SAP Binary Protocol XML is an internal binary SAP variant of asXML. It can only be used in the SAP system environment. With binary and structure compression however, it provides considerable performance and storage advantages compared to other XML options.
  The XML is generated using the CALL TRANSACTION command: http://ld9460.wdf.sap.corp:50000/sap/public/bc/abap/docu?sap-language=EN&sap-client=000&format=STANDARD&object=ABENABAP_XML_TRAFOS&tree=X

7. If you do not want to use the default system code page (UTF-8), specify a different one.

4.12.4 Edit Fields

You define the fields to be transferred together with their properties.

Context

If you have created the open hub destination based on a template, you will be offered all fields from the template as fields for the open hub destination. You can edit the field list by removing or adding fields. You can also change the properties of these fields.

If the template for the open hub destination is a DataSource, the field SOURSYSTEM is automatically added to the field list with reference to InfoObject 0SOURSYSTEM. This field is required if data from heterogeneous source systems is being written to the same database table. The data transfer process inserts the source system ID that is relevant for the connected DataSource. You can delete this field if it is not needed.

If you have selected *Database Table* as the destination and *Semantic Key* as the property, the field list is given an additional column. You can define the key fields for the semantic key here.

Procedure

1. Choose the *Details* tab.
2. To add an InfoObject, choose *Add InfoObject*. Specify the start of the name, select an InfoObject from the result list, and press *OK*. You can also select multiple InfoObjects by pressing *Ctrl + Shift*. The InfoObjects are added to your open hub destination. Any compound characteristic are added too.
3. You can now configure further settings for the InfoObject properties in the right screen area.
4. To add a field, choose *Add Field*. The field is added to your open hub destination. You can now define the field in the right area of the screen.
5. Enter a description for the field.
6. You can specify an InfoObject under *Identify with:*. This changes the field into an InfoObject. The properties of the InfoObject are applied. You can undo this by switching to *Field* under *Identify with:*.
7. Specify a data type and data length for the field.
8. To delete InfoObjects or fields, choose *Delete Field(s)*.
9. Under *Manage Keys*, you can define keys for the open hub destination.

4.13 Working with Source Systems

You can create source systems from the BW Modeling tools and can call various source system functions.

- In the *Data Sources* tree in the *Project Explorer* view, you can create and edit source systems from the context menu in the relevant folder of the source system type.
- In a source system's context menu, you can specify whether empty folders in the hierarchy of the source objects (application component hierarchy) should be shown or hidden.
- For replicating source systems, you can replicate the DataSources via the context menu.

Related Information

Creating a Source System [page 455]
Hiding Empty Application Components [page 466]
Executing Replication [page 468]

4.13.1 Creating a Source System

To connect a source for data access and data transfer with SAP BW/4HANA, you create a source system.

Prerequisites

Prerequisites for Creating a Source System [page 456]

Procedure

1. You are in the BW Modeling tools. Here you can create a new source system using the function *New* > *Source System*:
   - from the *File* menu
   - from the context menu of your BW project
   - via the context menu for the *Data Sources* node of your project
   - Via the context menu for the folder of the required source system type in the *Data Sources* tree

   More Information: Creating a Source System Using the Wizard [page 460]
2. Perform the remaining steps to create the source system.

More information:

- Creating a Source System of Type SAP HANA Local Database Schema [page 460]
- Creating a Source System of Type SAP HANA Smart Data Access [page 461]
- Creating a Source System of Type SAP HANA Tenant Database Schema [page 462]
- Creating Source Systems of Type “Big Data” [page 463]
- Creating Source Systems of Type ODP (RFC) [page 464]
- Creating Source Systems of Type ODP (SOAP/HTTP) [page 465]
- Creating Source Systems of Type “File” [page 466]

3. Once you have created the source system, go to the *Project Explorer* view in the *Data Sources* tree in the folder context menu for the source system type, and choose *Refresh*.

Results

You have created a source system connection to the source. In the *Project Explorer* view, you will find the corresponding source system folder in the *Data Sources* tree. The source systems are grouped here into separate folders for each source system type. In the case of ODP source systems and SAP HANA source systems, they are grouped for each ODP context or connection type.

To group the DataSources in the source system folder, either the hierarchical grouping of the source is used as the application component hierarchy, or the generic application component hierarchy for the non-replicated source systems is used. In the default setting, empty application components are also displayed.

You can now create DataSources. For SAP HANA source systems, it is also possible to create Open ODS views.

Related Information

Application Component Hierarchy [page 185]

4.13.1.1 Prerequisites for Creating a Source System

Depending on the source system type, different prerequisites apply.

4.13.1.1.1 Prerequisites for Creating a SAP HANA Source System

Depending on the connection type, different prerequisites apply for creating a SAP HANA source system.

Connection type *Local SAP HANA Database Schema*

The connection type uses the default connection to the SAP HANA database. The SAP <SID> user therefore requires the following authorizations: Object privilege SELECT on schema or object

Connection type *SAP HANA Smart Data Access*:

To use SAP HANA Smart Data Access, you have to configure the following system requirements and prerequisites:

- The required data provisioning adapters have been set up.
- The required remote sources have been created.
- The connection type uses the default connection to the SAP HANA database. The SAP <SID> user therefore requires the following authorizations:
  - Object privilege CREATE VIRTUAL TABLE
  - If you want to use real-time replication: Object privilege CREATE REMOTE SUBSCRIPTION
- To be able to use SAP HANA Smart Data Integration, the following additional system requirements also have to be met:
  - The minimum requirement for SAP HANA is Support Package Stack (SPS) 10. Note, however, that a large number of adapters have been added with SAP HANA SPS 11. For more information, see the Product Availability Matrix for SAP HANA smart data integration.
  - The SAP HANA smart data integration option is imported to SAP HANA.
  - The required data provisioning agenten have been installed.

Connection type *SAP HANA Tenant Database Schema*

The connection type uses the default connection to the SAP HANA database. The SAP <SID> user therefore requires the following authorizations: Object privilege SELECT on schema or object

For more information about prerequisites and supported database objects, see SAP Note 2312583. - FAQ. BW and SAP HANA Tenant Database: Cross-Database Access in SAP HANA Tenant Databases.

4.13.1.1.2 Prerequisites for Creating a Big Data Source System

There are various prerequisites for creating a big data source system.

- A remote source is available, which you can use to define a connection to a big data source using a SPARK SQL (DESTINATION) or VORA (ODBC) adapter.
  For more information, see *SAP HANA Hadoop Integration* and *SAP HANA Vora* in the *SAP HANA Administration Guide* on http://help.sap.com/hana_platform.
- The source system uses the default connection to the SAP HANA database. The SAP <SID> user therefore requires the object privilege CREATE VIRTUAL TABLE.

4.13.1.1.3 Prerequisites for Creating an ODP Source System

Various prerequisites apply for creating an ODP source system. These depend to some extent on the ODP context or the SAP repository to be connected.

- If you want to connect an SAP Business ByDesign system: Make sure that you have the required information and that the required settings have been made in the SAP BW/4HANA system.
  More information: Notes About Transferring Data from SAP Business ByDesign Systems Using ODP [page 175]
- If you want to connect SAP S/4HANA Cloud: Make sure that secure communication between SAP S/4HANA Cloud und SAP BW/4HANA is set up in SAP S/4HANA Cloud using communication administration.

  > **Note**
  >
  > For information about the availability of ABAP CDS views for extraction from SAP S/4HANA Cloud to SAP BW, see SAP Note 2674373.

  More information: Extracting CDS Views to SAP BW/4HANA in the documentation for SAP S/4HANA Cloud
- If you want to connect a source using SAP Landscape Transformation Replication Server (SLT): Make sure that the connection to the data source has been set up in the SLT.
  More Information: Transferring Data from SLT Using Operational Data Provisioning [page 171]
- For all ODP contexts and SAP repositories:
  - You have made the following settings in the SAP BW/4HANA Customizing Implementation Guide, under *SAP BW/4HANA* > *Data Collection* > *Links to Other Source Systems*.
    - General connection settings
    - Perform automatic workflow Customizing
  - As a rule, system changes are not permitted in production systems. However, connecting a system as a source system to a SAP BW/4HANA, or connecting a SAP BW/4HANA to a new source system, results in modifications to the system change options. In the client used in the SAP BW/4HANA system, you have therefore ensured that the following changes are permitted during the source system connection:
    - Cross-client Customizing and repository changes
      In the SAP BW/4HANA Customizing Implementation Guide, under *SAP BW/4HANA* > *Data Collection* > *Links to Source Systems* > *General Connection Settings* > *Assign Logical System to*

Client, select the client in question and choose *Goto* > *Details*. In the *Cross-Client Object Changes* field, select *Changes to Repository and Cross-Client Customizing Allowed*.

- Changes to the local developments and Business Information Warehouse software components
  You use transaction SE03 (Organizer Tools) to set the change options. Choose *Organizer Tools* > *Administration* > *Set System Change Option*. Choose *Execute*. On the next screen, make the following settings:.
- Changes to the customer name range
  You also use transaction SE03 to set the change option for the customer name range.
- Changes to BW namespaces /BWC/ and /BW0/
  You also use transaction SE03 to set the change option for the BW namespaces.

4.13.1.1.4 Prerequisites for Creating a File Source System

There are various prerequisites for creating a file source system.

- You have made the following settings in the SAP BW/4HANA Customizing Implementation Guide, under *SAP BW/4HANA* > *Data Collection* > *Links to Other Source Systems*.
  - General connection settings
  - Connection between flat files and SAP BW/4HANA system
  - Perform automatic workflow Customizing
- As a rule, system changes are not permitted in production systems. However, connecting a system as a source system to a SAP BW/4HANA, or connecting a SAP BW/4HANA to a new source system, results in modifcations to the system change options. Therefore, in the client used in the SAP BW/4HANA system, make sure that the following changes are permitted during the source system connection:
  - Cross-client Customizing and repository changes
    In the SAP BW/4HANA Customizing Implementation Guide, under *SAP BW/4HANA* > *Data Collection* > *Links to Source Systems* > *General Connection Settings* > *Assign Logical System to Client*, select the client in question and choose *Goto* > *Details*. In the *Cross-Client Object Changes* field, select *Changes to Repository and Cross-Client Customizing Allowed*. *Customizing allowed*.
  - Changes to the local developments and Business Information Warehouse software components
    You use transaction SE03 (Organizer Tools) to set the change options. Choose *Organizer Tools* > *Administration* > *Set System Change Option*. Choose *Execute*. On the next screen, make the following settings:.
  - Changes to the customer name range
    You also use transaction SE03 to set the change option for the customer name range.
  - Changes to BW namespaces /BIC/ and /BI0/
    You also use transaction SE03 to set the change option for the BW namespaces.

4.13.1.2 Creating a Source System Using the Wizard

You will now be guided through the creation process by a wizard. In the wizard, you specify general specifications for the technical name and description, and select the source system type. The other steps you see in the wizard and editor depend on the source system type that you have selected.

You are in the wizard for creating the source system.

Procedure

1. In the first wizard step *New Source System*, enter a technical name for the source system in *Name* The system checks the name of the source system. The name can be up to 10 characters long.
2. Enter a *Description* for the source system.
3. Choose *Next*.
4. In the next wizard step, select the source system type. If have created the source system from the folder of the source system type in the *Data Sources* tree, the correct type is preselected.
5. Choose *Next*.
   The next wizard steps depend on which type you selected.
   More Information:
   - Creating a Source System of Type SAP HANA Local Database Schema [page 460]
   - Creating a Source System of Type SAP HANA Smart Data Access [page 461]
   - Creating a Source System of Type SAP HANA Tenant Database Schema [page 462]
   - Creating Source Systems of Type “Big Data” [page 463]
   - Creating Source Systems of Type ODP (RFC) [page 464]
   - Creating Source Systems of Type “File” [page 466]

Results

Once you have run all wizard steps, the source system editor opens where you can finish creating the source system.

4.13.1.2.1 Creating a Source System of Type SAP HANA Local Database Schema

The SAP HANA source system provides central and unified access to tables, views, and native DataStore objects in the local SAP HANA database or in a SAP HANA tenant database schema, and to sources that are provided using SAP HANA Smart Data Integration and SAP HANA Smart Data Access. Use these wizard steps to select the connection parameters for the SAP HANA source system of type *SAP HANA Local Database Schema*.

Prerequisites

More Information: Prerequisites for Creating a SAP HANA Source System [page 457]

Procedure

You have opened the wizard for creating source systems and you have used the first wizard step to specify the technical name, description and type of the source system.

1. Choose *Next*.
2. In the wizard step *Select a DB Object Schema*, enter a search term and, in the list of search results, select the schema you want to create.
3. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
4. Choose *Finish* to return to the source system editor.
5. [Optional:] Under *General Properties*, specify the business content type and release.
6. Save and activate the source system.

4.13.1.2.2 Creating a Source System of Type SAP HANA Smart Data Access

The SAP HANA source system provides central and unified access to tables, views, and native DataStore objects in the local SAP HANA database or in a SAP HANA tenant database schema, and to sources that are provided using SAP HANA Smart Data Integration and SAP HANA Smart Data Access. Use these wizard steps to select the connection parameters for the SAP HANA source system of type *SAP HANA Smart Data Access*.

Prerequisites

More Information: Prerequisites for Creating a SAP HANA Source System [page 457]

Procedure

You have opened the wizard for creating source systems and you have used the first wizard step to specify the technical name, description and type of the source system.

1. Choose *Next*.
2. In the wizard step *Select a remote source*, enter a search term. Then use the list of search results to select the required remote source, which defines the connection to the source.
3. Choose *Next*.

Depending on the remote source and the adapter, which is specified in the remote source, the system displays following wizard steps:
For fields whose objects are in schemas:

1. In the wizard step *Select a DB Object Schema*, enter a search term and, in the list of search results, select the schema you want to create.

For sources with a hierarchical folder structure, you can select a path for a given folder for which you want to create the source system from the folder structure:

1. In the wizard step *Select Path*, enter a search term and select the path prefix from the hierarchy displayed.

4. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
5. Choose *Finish* to return to the source system editor.
6. [Optional:] Under *General Properties*, specify the business content type and release.
7. Save and activate the source system.

4.13.1.2.3 Creating a Source System of Type SAP HANA Tenant Database Schema

The SAP HANA source system provides central and unified access to tables, views, and native DataStore objects in the local SAP HANA database or in a SAP HANA tenant database schema, and to sources that are provided using SAP HANA Smart Data Integration and SAP HANA Smart Data Access. Use these wizard steps to select the connection parameters for the SAP HANA source system of type *SAP HANA Tenant Database Schema*.

Prerequisites

More Information: Prerequisites for Creating a SAP HANA Source System [page 457]

Procedure

You have opened the wizard for creating source systems and you have used the first wizard step to specify the technical name, description and type of the source system.

1. Choose *Next*.
2. In the wizard step *Select a Remote Database*, enter a search term and select the remote database from the list of search results.
3. Choose *Next*.
4. In the wizard step *Select a DB Object Schema*, enter a search term and, in the list of search results, select the schema you want to create.
5. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.

6. Choose *Finish* to return to the source system editor.
7. [Optional:] Under *General Properties*, specify the business content type and release.
8. Save and activate the source system.

4.13.1.2.4 Creating Source Systems of Type “Big Data”

The Big Data source system enables central, unified access to sources, which can be provided via SAP HANA Smart Data Access by using the SPARK SQL (DESTINATION) or VORA (ODBC) adapters. Use these wizard steps to select the connection parameters for the Big Data source system.

Prerequisites

More Information: Prerequisites for Creating a Big Data Source System [page 458]

Procedure

You have opened the wizard for creating source systems and you have used the first wizard step to specify the technical name, description and type of the source system.

1. Choose *Next*.
2. In the wizard step *Select a remote source*, enter a search term. Then use the list of search results to select the required remote source, which defines connection to the Big Data source via the SPARK SQL (DESTINATION) or VORA (ODBC) adapters.
3. Choose *Next*.
4. In the wizard step *Select a DB Object Schema*, enter a search term and, in the list of search results, select the schema you want to create.
5. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
6. Choose *Finish* to return to the source system editor.
7. [Optional:] Under *General Properties*, specify the business content type and release.
8. Save and activate the source system.

4.13.1.2.5 Creating Source Systems of Type ODP (RFC)

If you want communication between the source and SAP BW/4HANA via RFC, define the destination and specify how the DataSources in the source system should be grouped.

Prerequisites

More Information: Prerequisites for Creating an ODP Source System [page 458]

Procedure

You have opened the wizard for creating source systems and you have used the first wizard step to specify the technical name, description and type of the source system.

1. Choose *Next*.
2. In the *ODP Page Selection* wizard step, configure how the source should be connected with the system:
   - *RFC with Remote System*: A remote system should be connected with the SAP BW/4HANA system by RFC.
   - *RFC in the Same BW System*: The RFC destination should point to the SAP BW/4HANA system itself (myself scenario).
3. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
4. Choose *Finish* to return to the source system editor.
5. [Optional:] Under *General Properties*, specify the business content type and release.
6. Under *Specific Properties*, create the destinations to the source system. You require a logical destination from the SAP BW/4HANA system to the source, together with a callback destination. The dialog destination is optional.
   1. For *RFC with Remote System*, the following applies: The system has created an RFC destination with a dummy target host `<ENTER SERVER NAME>`.
      1. To maintain the destination correctly, select the logical destination in the editor under *Specific Properties* and choose *Edit*. You access RFC destination maintenance (transaction SM59).
      2. Enter the required data for the RFC destination.
      3. Save and test the destination.
      4. Return to the editor for creating the source system. Here the system displays the logical destination and callback destination as valid destinations.
   2. For *RFC in the Same BW System*, the following applies: The logical destination already exists and is valid.
7. To select an ODP context, choose *Specific Properties* > *ODP Context*. Note that this is only possible once a valid logical destination is available.
8. Under *Specific Properties* > *Remote Tree*, you can specify for specific sources, which can deliver a hierarchical grouping of source objects (tree), whether the grouping should be copied to the DataSource tree for the source system.

If you do not copy the tree from the source, or the source cannot deliver a hierarchical grouping, the generic application component hierarchy defined for the non-replicated source systems is used as the grouping for the DataSource tree.
If the *Remote Tree* field is selected, you can update the hierarchical grouping from the source by pressing *Upload Tree* in the editor toolbar.

9. Save and activate the source system.

4.13.1.2.6 Creating Source Systems of Type ODP (SOAP/HTTP)

If communication between the source and SAP BW/4HANA via SOAP/HTTP is required, creating the source system involves the following steps: You perform Web Service configuration in the SOA Manager, select the relevant ODP context, and specify how the DataSources are to be grouped in the source system.

Prerequisites

More Information: Prerequisites for Creating an ODP Source System [page 458]

Procedure

You have opened the wizard for creating the source system, used the wizard to specify a technical name and a description, and selected the source system type.

1. Choose *Next*.
2. In the wizard step *ODP Page Selection*, select *SOAP/HTTP*.
3. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
4. Choose *Finish* to return to the source system editor.
5. [Optional:] Under *General Properties*, specify the business content type and release.
6. Choose *Specific Properties* > *SOA Manager* to call Web service configuration in the SOA Manager.
   In the Web service configuration, create and configure a logical port for the default consumer proxy CO_RSDSPX_ODP_IN. Note that the logical port must have the same name as the source system. Check and save the consumer proxy and then go back.
   More information: Configuring a Service Consumer
7. Choose *Specific Properties* > *ODP Context* to selecte an ODP context.
   You can use the contexts ABAP_CDS (for ABAP Core Data Services Views) and BYD (for SAP Business ByDesign).
8. Under *Specific Properties* > *Remote Tree*, you can specify for specific sources, which can deliver a hierarchical grouping of source objects (tree), whether the grouping should be copied to the DataSource tree for the source system. Note that the remote tree is only supported for certain contexts, for example ABAP_CDS.

If you do not copy the tree from the source, or the source cannot deliver a hierarchical grouping, the generic application component hierarchy defined for the non-replicated source systems is used as the grouping for the DataSource tree.
If the *Remote Tree* field is selected, you can update the hierarchical grouping from the source by pressing *Upload Tree* in the editor toolbar.

9. Save and activate the source system.

4.13.1.2.7 Creating Source Systems of Type “File”

For source systems of type *File*, you only need a technical name and a description.

Prerequisites

More Information: Prerequisites for Creating a File Source System [page 459]

Procedure

You have opened the wizard for creating source systems and you have used the first wizard step to specify the technical name, description and type of the source system.

1. [Optional:] To display an overview of the settings you have made in previous wizard steps in a following wizard step, choose *Next*.
2. Choose *Finish* to return to the source system editor.
3. [Optional:] Under *General Properties*, specify the business content type and release.
4. Save and activate the source system.

4.13.2 Hiding Empty Application Components

In the default setting, application components without DataSources are displayed in the Data Sources tree of a BW project in the BW Modeling tools. To have a better overview, you have the option of hiding empty application components.

Procedure

You are in the BW Modeling tools in the *Project Explorer* view. For a source system, the *Show Empty Folders* function is selected by default in the context menu.

Choose *Show Empty Folders* from the source system’s context menu.

Results

The system hides all application components that do not contain any DataSources from the application component hierarchy. To be able to create a DataSource in the hidden application component, you have to show the application components again by choosing *Show Empty Folders*.

If there are no DataSources for the source system however, the application component hierarchy is still displayed in its entirety so that you can create a DataSource at all.

4.13.3 Creating and Moving Application Components in a Generic Hierarchy

For replicating source systems, application components are copied over from the source systems with the DataSources. For non-replicating source systems, you can create application components in the generic application component hierarchy yourself in the SAP BW/4HANA system and move components in the hierarchy.

Procedure

1. You are in the BW Modeling tools. In the Data Sources tree on your source system, choose *New* > *Application Component*. The wizard opens.
2. Enter a technical name and a description.
3. When you have finished making your settings, choose *Finish*. The editor appears. You can change the description. The source system is not displayed here because the application component does is not dependent on the source system.
4. You can move the application component by choosing *Move* in the context menu in your application component, or by using drag and drop. Specify the position in the hierarchy where you want to insert the application component: as a subnode, above or below the target application component.

   > ⓘ Note
   >
   > If you have activated alphabetical sorting for BW objects, you cannot choose whether the application component should be inserted above or below.

5. Choose *Add to Favorites* to add the application component to your favorites. Since the application component is used for different source systems, the source system - for which you have added the application component to the favorites - is shown in parentheses after the application component.

4.13.4 Executing Replication

For replicating source systems, you can replicate the DataSources in the BW modeling tools.

You are in the *Project Explorer* view in the BW Modeling tools.

1. In the context menu for the source system or the application component that you want to replicate the DataSources for, choose *Replicate*. You can also replicate an individual DataSource from the context menu for a DataSource.
   The system collects the DataSources and displays them in the *Replicate DataSources* wizard.
2. In the default setting, all DataSources are selected to be added to the next wizard step.
3. Use the *Display* field to see which DataSources are affected by the replication, and how:

Value in the *Display* field	Description
all	The wizard displays all DataSources that can be replicated.
Create	The Wizard displays DataSources that were thus far not known in SAP BW/4HANA and that can be created during replication in SAP BW/4HANA.
Delete	The Wizard displays DataSources that no longer exist in BW and that can be deleted during replication there. **Note** The DataSource is needed for other objects, such as transformation or the data transfer process. Therefore, make sure that the DataSources, which are selected using this setting, really need to be deleted.
Equal To	The wizard displays DataSources with no differences in the source system and SAP BW/4HANA. The DataSources are updated during replication in SAP BW/4HANA.
Update (compatible)	The wizard displays DataSources that there are compatible changes for in the source system. The DataSources are updated during replication in SAP BW/4HANA.
Update (incompatible)	The wizard displays DataSources that there are incompatible changes for in the source system. The DataSources are set to inactive during replication in SAP BW/4HANA and must be adapted in the DataSource editor using the update function.

4. Specify which DataSources should be taken into account during replication. If for example only one or a small number of DataSources need to be created, then these can be selected here.
   You can select/deselect the DataSources individually with the *Select Displayed DataSources* and *Deselect Displayed DataSources* pushbuttons, and can select/deselect the DataSources displayed in the Wizard for all DataSources depending on the selection made under *Display*.

5. Choose *Next*.
6. In the next wizard step, the system shows how many DataSources have been created, updated or deleted, and how many DataSources have been set to inactive.
   You can specify here whether to activate updated DataSources in the SAP BW/4HANA system (the default setting) and whether to activate newly created DataSources (this must be set by the user).
7. Choose *Finish*.

Result

The replication is started as a background job in the SAP BW/4HANA back end. In the BW Modeling tools, the job log is displayed in the Editor area.

The replication is initially performed without package assignment and without transport connection. The replicated objects must subsequently be written to packages and to a transport request.

4.13.5 Set Source System to Inactive

Query access using an Open ODS view or a CompositeProvider with an affected Open ODS view leads to an error when the source that is connected with the source system to the SAP BW/4HANA system is not available. To avoid this happening in particular for queries on CompositeProviders, you can set the object status for source systems to inactive. For inactive source systems, query access using the Open ODS view or CompositeProvider is suppressed.

Systems that are connected with source systems to SAP BW/4HANA can have different Service Level Agreements or maintenance plans to the SAP BW/4HANA system. The operation of these systems can be decoupled from the SAP BW/4HANA system. For this reason a source may not be available if a query directly accesses data in the source.

Open ODS views allow direct query access to sources. So, if a source is not available, a query on an Open ODS view results in an error.

Queries on CompositeProviders that contain Open ODS views as participating InfoProviders are also affected by this. In the worst case, all queries of a CompositeProvider result in an error if the source for one of the participating Open ODS views is not available.

Suppress Read Acess Temporarily: Set Object Status of Source System to Inactive

1. In the BW Modeling tools, open the editor of the relevant source system.
2. In the toolbar of the source system editor, choose *Deactivate Source System*.

For Open ODS views that use this source system, query access is suppressed. This means that no data is displayed, only an appropriate message.

Neither is data displayed for Open ODS views that are part of a CompositeProvider and use this source system. The CompositeProvider only displays data for other InfoProviders, such as DataStore objects.

Inactive source systems also affect data transfer processes with DataSources of this source system. For inactive source systems, no data is loaded. Instead, the system displays an appropriate message.

> **Note**
>
> Note that suppression of query access only takes place if the query was defined directly on the Open ODS view, or the Open ODS view is used in a CompositeProvider with Union. For an Open ODS view with master data semantic that is used by another Open ODS view or CompositeProvider through an association for navigation attributes, suppression does not work. For CompositeProviders with a join scenario, suppression is not supported.

Reactivate Read Access: Set Object Status of Source System to Active

When the source is available again, you can set the source system to active again.

1. In the BW Modeling tools, open the editor of the relevant source system.
2. In the toolbar of the BW Modeling tools, choose  *Activate BW Object*.

4.14 Working with DataSources

You can create and edit DataSources in the BW Modeling tools.

- In the *Data Sources* tree in the *Project Explorer* view, you can create and edit DataSources via the context menu for a source system or the application component of the source system. You can create DataSources using the guidance of a wizard and specify the DataSource properties in a native Eclipse editor.
- For replicating source systems, you can replicate the DataSource in the *Data Sources* tree in the *Project Explorer* view via the context menu.

Related Information

Creating DataSources [page 471]
Editing DataSources [page 496]
Metadata Comparison and Updating DataSources [page 497]
Executing Replication [page 468]

4.14.1 Creating DataSources

To be able to access data from a source, and to transfer this data to the system, the metadata of the source object must be available the system in the form of a DataSource. In addition to replicating the DataSource from a replicating ODP source system, you can also create a DataSource in the BW/4HANA system manually using the DataSource Wizard.

Context

You create DataSources using a wizard and define the DataSource properties in a native Eclipse editor.

Procedure

1. You are in the BW Modeling tools. In the context menu for your BW project, choose *New* > *DataSource*. Alternatively,you can create DataSources in your BW project's *Data Sources* tree or from the context menu on the source system or on the application component of a source system by choosing *Copy* in the context menu of an existing DataSource or a data flow.

   You will be guided through the process by a wizard. More information: Creating a DataSource Using the Wizard [page 472]

   Once you have completed all the steps in the wizard, the DataSource editor opens.
2. Perform the remaining steps to create the DataSource:
   a. Define the general properties on the *Overview* tab.
   b. On the *Extraction* tab, you make extractor-specific settings.

      For SAP HANA DataSources and ODP DataSources, you can exchange the source object. If you want to exchange the source object of the DataSource, press the *Exchange* button and then select the new source object. The *Confirm Update DataSource* opens. This takes an initial proposal from the back end to update the DataSource, and displays this in comparison with the current DataSource version.
   c. On the *Fields* tab, you maintain the properties of the DataSource fields.

      If you select a field in the field list, you can maintain the field properties on the right of the DataSource editor. You define the language-dependency or time-dependency of a field in the field list via the field's context menu.

      If you select multiple fields in the field list, you can use mass maintenance to change the properties of the selected fields. If the selected fields have different values for a particular property, this is indicated by the expression <multiple values> or the corresponding flagging of a checkbox. A change made to a property applies to all selected fields. If a property cannot be changed for one of the selected fields, the property cannot be changed in mass maintenance. If the selected fields have different data types, no information is displayed for the length, the number of digits, and the number of decimal places.
3. Check, save and activate the DataSource.
4. By choosing *Show Data Preview* in the editor toolbar or *Window* > *Show View* > *Others*, you can call the *Data Preview for DataSource* view. In this view, the system displays a data preview for the active version

of the DataSource. Here you can specify how many data records to display in the preview. For hierarchy DataSources, you can select which segment to the data is displayed for.

5. Once you have created the DataSource, choose *Refresh* in the context menu of the source system or the application component.

   The DataSource is now displayed in the *Data Sources* tree in the application component of the source system and is ready for use.

Related Information

Creating DataSources for ODP Source Systems [page 487]
Creating an SAP HANA DataSource [page 475]
Creating a Big Data DataSource [page 486]
Creating DataSources for File Source Systems [page 488]
Executing Replication [page 468]
Metadata Comparison and Updating DataSources [page 497]

4.14.1.1 Creating a DataSource Using the Wizard

You will be guided through the process by a wizard. In the wizard, you specify what the new DataSource will be based on. You can enter a name, description, DataSource type and - if applicable - specify the assignment to the application component.

Procedure

In the *Project Explorer* view, you have chosen *New* > *DataSource* from the context menu of your BW project. You are in the wizard for creating the DataSource and your are on the first wizard step *New DataSource*.

1. Select the source system.

   Once you have made a selection, the wizard displays the description and the source system type.
2. Choose *Next*.
3. In the wizard step *Select Template or Source object*, select the template or source that you want your new DataSource to be based on. Then choose *Next*.

   There are various options available, depending on the source system type:

Template or Source Object	Source System Type SAP HANA and Big Data	Source System Type ODP	Source Source System File
None	-	-	x

Template or Source Object	Source System Type SAP HANA and Big Data	Source System Type ODP	Source Source System File
Copy from DataSource (Same Source System Only)	x	x	x
Proposal from Operational Data Provider	-	x	-
Proposal from SAP HANA Table or View	x	-	-
Proposal from SAP HANA Native DataStore Object	Possible for the following types of SAP HANA source systems: • SAP HANA Local Database Schema • SAP HANA Tenant Database Schema • SAP HANA Smart Data Access (with HANA Adapter or HANA ODBC Adapter)	-	-
DataSource as Template	-	-	x
InfoSource as Template	-	-	x
DataStore Object as Template	-	-	x
InfoObject as Template	-	-	x

If you selected *InfoObject as Template*, also specify if you want to use attributes, texts, or hierarchies of the InfoObject as a template.

You are taken to the next wizard step: *Select <object type of the source object or template>*.

If you have a file source system and if you have selected *None*, you are taken directly to the last wizard step: *Confirmation Page*.

4. In the *Select <Object Type of the Source Object or Template>* wizard step, select the object that you want to use as the basis for the DataSource you are creating, and choose *Next*.

> **Note**
>
> - If you selected *Template from DataSource* (for a file source system) in the previous wizard step, the system searches for the specified search string in the DataSource name, in the description and in the source system name.
> - For ODP source systems with context *ABAP_CDS*, note that the SQL view names of the ABAP CDS views are displayed in the search result list.

5. Enter the following information in the wizard step *Confirmation Page*:
    - *Application Component*: Select an application component.

      > **Note**
      >
      > If, while creating an ODP source system, you specified that the hierarchical grouping of the source should be applied as the application component hierarchy of the source system, you cannot specify any application components here. The application component is predefined by the template object.

    - *Add to Favorites*: We recommend selecting this field so that you can access the DataSource faster.
    - *Name*: Enter a technical name for the DataSource.
    - [Optional:] *Description*: Enter a description.
    - *DataSource Type*: Specify if you want to use the DataSource to access master data attributes, master data texts, hierarchies or transaction data.

      > **Note**
      >
      > If you create a file DataSource based on an InfoObject template, the type is set that you selected for the InfoObject in step 3. You can then no longer change the *Hierarchies* type. You can then no longer change the *Master Data Attributes* or *Master Data Text* type into the *Hierarchies* type.
      >
      > If you create a DataSource based on a DataStore object template or InfoSource template, you cannot select the *Hierarchies* type.
      >
      > If you create a DataSource based on the template of a hierarchy DataSource (from the local system or another system) to create, only the *Hierarchies* type is allowed.

6. Choose *Finish*.

Results

Once you have completed all wizard steps, the DataSource editor opens.

4.14.1.2 Creating an SAP HANA DataSource

Before you can transfer data from a SAP HANA source system, the metadata from the source in must be available in the SAP BW/4HANA system in the form of a DataSource.

Context

> **Note**
>
> Input parameters of SAP HANA views are not visible in the DataSource. This applies to both local views and views in the tenant database as well as for access using SAP HANA Smart Data Access. No values are given for the parameter when accessing the view. For more information, see SAP Note 2142344.

Procedure

1. You are in the BW Modeling tools. Here you open the context menu for your BW project and choose *New* > *DataSource* .
   The wizard for creating DataSources appears.
2. Make the required entries in the wizard and choose *Finish*. More Information: Creating a DataSource Using the Wizard [page 472]

   > **Note**
   >
   > If you have published the hierarchical structure of the remote source with Enterprise Semantic Services, the value help accesses the indexed data and is faster as a result. For more information about Enterprise Semantic Services, see *Enable Enterprise Semantic Services* in the *Administration Guide for SAP HANA Smart Data Integration and SAP HANA Smart Data Quality*.

   For smart data access sources with a hierarchical file structure: The tables and views are displayed in the search results list with the *Path Suffix* – the remaining part of the folder structure path up to the source object (excluding the source object) that is not defined in the source system.

   The DataSource editor appears.
3. On the *Overview* tab, you make general settings, if required.
4. On the *Extraction* tab, you make extractor-specific settings.
   a. [Optional:] You can exchange the source object of the DataSource here. Note that native DataStore objects can only be exchanged with other native DataStore objects, and tables/views can only be exchanged with other tables/views.
   b. For data provisioning adapters that support real-time replication: Under *Real-Time Replication*, you select the replication method by selecting the corresponding remote subscription types.
5. On the *Fields* tab, you maintain the properties of the DataSource fields.
   a. Specify which DataSource fields are to be used when reading from the source object (local table/view or virtual table) for packaging in the data transfer process and for the partitions of the replication table.

To do this, select the relevant field in the field list and select the *Use for Packaging and Partitioning* indicator in the field properties under *SDA-Specific Properties*. If you select properties for multiple fields, disjunctive selections are made across all fields selected.

For connection type *SAP HANA Smart Data Access (SDA)*, fields **must** be selected for packaging and partitioning to allow splitting of a data transfer process (DTP) over multiple packages. For other connection types, such as *Local SAP HANA Database Schema*, fields **can** be selected for packaging and partitioning. For more information, see *Packaged Data Transfer from an SAP HANA Source System*.

b. For real-time replication with remote subscription type INSERT: Specify which fields are to be treated like key figures that can be totaled. To do this, select the relevant field in the field list and select the *Key Figure (Summation)* indicator in the field properties under *SDA-Specific Properties*.

   If you want the generated SAP HANA view to reverse the sign of a field value for before and reverse images (`SDI_CHANGE_TYPEB` and `D`), choose *Key Figure (Summation)*. If you want a field to be interpreted like a characteristic, for example, like a numeric label for a person or a material, do not set the indicator.

> ### Note
>
> If you transfer data from an SAP HANA source system to a DataStore object with change log using the delta process *Complete delta update with deletion indicator* (ABR) and you cannot rule out the occurrence of before images without after images, choose the "Summation" aggregation type for all key figures in the transformation.

6. Save and activate the DataSource.
7. Once you have created the DataSource, choose *Refresh* in the context menu of the source system or the application component.

Results

The DataSource is displayed in the *Data Sources* tree in the application component of the source system and is ready for use.

When you activate the DataSource, the system generates various objects such as the transfer program. In the *Properties* view, you can display the generated objects on the *Technical Attributes* tab page.

If the connection to the source is established using SAP HANA smart data access, a view on the virtual table is generated for query access. In the same way, a view is created for local tables. The main task of this view is the type conversion of SAP HANA data types and ABAP data types. The view is used in the data preview for direct access and full requests if no remote subscriptions of type UPSERT are used and for delta requests if no remote subscriptions of any type are used.

If you use real-time replication, additional SAP HANA objects are generated, depending on the real-time replication method selected:

- Remote subscription(s)
- Target table(s) for the real-time replication
- SQL view(s) on the target table

In the *Properties* view, you can display the objects generated for SAP HANA smart data access and real-time replication on the *Extraction* tab page.

From the *Properties* view, you can call a generated object by clicking the mouse.

Related Information

Creating a DataSource Using the Wizard [page 472]
Real-Time Replication (controlled by SAP BW/4HANA) [page 148]
Remote Subscription of Type UPSERT [page 149]
Remote Subscription of Type INSERT [page 150]
Condition for Static Filters for SAP HANA DataSources [page 158]
Packaged Data Transfer from an SAP HANA Source System [page 477]
Key Figure Summation for Before Images Without After Images [page 152]
Administration Guide for SAP HANA Smart Data Integration and SAP HANA Smart Data Quality
SAP Note 2142344

4.14.1.2.1 Packaged Data Transfer from an SAP HANA Source System

In the SAP HANA DataSource, you determine which fields of the DataSource to use for packaging in the data transfer process, as long as no grouped extraction is defined in the transformation or in the data transfer process.

Background

A data transfer process generally divides the data in a request into multiple packages, to ensure that transformation (and further processing) is carried out in a resource-efficient manner. In particular, this ensures that sufficient system resources are available for transformation (and further processing) (for example, there is no memory overflow). You configure the package size in the data transfer process.

Special Feature of SAP HANA Source Systems

A data transfer process that reads data from an SAP HANA source system uses disjunctive selections on specific columns to create package of the configured size. Which columns are used for this is something you define as a property of the fields of a DataSource, as long as no grouped extraction is defined in the transformation or in the data transfer process. For packaging and partitioning, choose columns that have neither a particularly high nor a particularly low cardinality. If, about 10,000 records are added to a source object per month for example, the calendar month (including year) would be a suitable column for packaging and partitioning. If you select multiple columns, disjunctive selections are made across all columns selected. In this case, you should ensure that the correlation between the selected columns is as low as possible.

Note the following in connection with the various connection types:

- For connection type *SAP HANA Smart Data Access (SDA)*, fields in the DataSource, the transformation, or the data transfer process **must** be selected for packaging and partitioning to allow splitting of a data transfer process (DTP) over multiple packages. If no fields are selected, all data in a request is transferred in a single package. In the case of larger amounts of data, the main memory might not have sufficient capacity. The DataSource check therefore issues a warning if no fields are selected for packaging and partitioning. Note that the fields for packaging and partitioning for real-time replication are also used for delta initialization and for the target table.
- For other connection types, such as *Local SAP HANA Database Schema*, fields **can** be selected for packaging and partitioning. If fields are selected, a DTP builds disjunct selections (like with SDA) across all selected fields, to divide up a request into packages. If no fields are defined, the data is divided up using a different algorithm wherever possible. If a generic BW delta process is configured and the delta field is a key field of the source object, the delta field is used for packaging. This algorithm is also applied if the source object does not contain any key fields, such as a database view. For tables in the column store of the SAP HANA database, an internal algorithm for packaging is applied if the delta field is not used and no other fields are selected. There are two situations where it would be better to have no fields selected for packaging:
  1. An outbound transformation requires the data to be sorted by the delta field of the generic BW delta process - not only within packages but also beyond packages.
  2. The internal algorithm for source tables in the column store of the SAP HANA database is more resource-efficient than packaging by selected fields.

  Therefore the DataSource check displays a warning if the prerequisites for one of the two situations have been met. The best option for a specific case depends on the outbound transformation.
  If one of the following statements applies to all outbound transformations, there is no need to sort the data by delta field:
  - The delta field is not a key field of the source object.
  - The delta field is key field of the data target of the transformation.
  - Every selected field for packaging is a key field of the data target of the transformation.
  - The aggregation “summation” is selected for all non-key fields of the data target of the transformation.

  For special transformations, especially with start routines, expert routines or end routines, it can be particularly advantageous or necessary in the transformation or the data transfer process to select specific fields for packaging, so that every value combination is transferred via these fields in exactly one package. However, it is generally more resource efficient (for source tables in the column store of the SAP HANA database) not to select any fields, so that the internal algorithm can be applied for packaging.

Special Features

If a data transfer process splits the data from a SAP HANA source system over several packages, the data is only read using a database cursor in certain cases. All packages of a request are therefore now only extracted in the same read transaction in certain cases. The read consistency therefore now only corresponds to the "Read Committed" isolation level on the SAP HANA database in certain cases. In general, the packages are extracted in different processes and therefore in independent read transactions. As a result, the read consistency generally corresponds to the "Read Uncommitted" isolation level. If write transactions are completed on the read data between the read transactions of the DTP request, "dirty reads" can occur.

Procedure for Avoiding Inconsistencies When Reading Packaged Data

If you have sufficient system resources in a specific case, you can configure the DTP to process all of the data in one package. If you want or require multiple packages, you can avoid the inconsistencies described by ensuring that the data set selected does not change during the independent read transactions. It is important to differentiate between the following cases:

1. Reading from a local target table for real-time replication (remote subscription type UPSERT or INSERT): In this case, the system automatically ensures consistent read behavior using the strictly monotonously increasing modification timestamp. Note that this option is available for some data provisioning adapters only.
2. Reading of initial data or data changes (generic BW delta) from a source object: In this case, you can ensure that the selected data set does not change during the independent read transactions by specifying an appropriate upper safety interval.
3. Reading of the current data (full) from a local database object: In this case, there is no generally valid approach. Check whether it is possible to waive the "read committed" isolation level or whether appropriate administrative measures can be employed to ensure that the selected data does not change (locks, time of day, or similar).

Note that in the second and third case, it is irrelevant whether your SAP HANA source system points to a local database schema or to a source (remote source using SAP HANA smart data access).

4.14.1.2.2 Native DataStore Object as Source

If the SAP HANA system, which is connected as a source system to SAP BW/4HANA, contains SAP HANA Data Warehousing Foundation 2.0 SPS01 or higher, the native DataStore object is available as a source object type of the SAP HANA source system. For native DataStore objects without a change log, the DataStore adapter supports query-like-access (open ODS view or full extraction via the data transfer process). In addition, delta extraction is supported for native DataStore objects with change logs.

The native DataStore object is supported as a source object for SAP BW/4HANA for the following connection types of the SAP HANA source system:

- SAP HANA Local Database Schema
- SAP HANA Tenant Database Schema
- SAP HANA Smart Data Access
  The remote SAP HANA system is connected to SAP BW/4HANA via the HANA ODBC adapter (Smart Data Access) or via the HANA data provisioning adapter (Smart Data Integration). In addition, SAP HANA Data Warehousing Foundation 2.0 SPS01 or higher must be installed on the remote SAP HANA system.

Prerequisites

In addition to the table for the active data and the change log, the native DataStore object contains multiple administration tables, which are used during data extraction. The following authorizations are therefore required:

- User SAP<SID> requires the INSERT, UPDATE and DELETE authorization for the administration tables in the native DataStore object in order to allow read and write access using the SAP BW/4HANA system's default database connection.
- In the case of SAP HANA Smart Data Access: The connection properties for the HANA ODBC in *DML Mode* must be set to `readwrite`. The user specified in the remote source definition must have the INSERT, UPDATE and DELETE authorization for the administration tables in the native DataStore object.

Modeling the DataSource

SAP HANA DataSources are created using the DataSource wizard. For the above-mentioned connection types of the SAP HANA source system, you can specify native DataStore objects as source objects here. With the exception of the delta extraction settings (see below), DataSource modeling is essentially the same as for tables or views as source objects. Note however that native DataStore objects as source objects in the DataSource editor can only be exchanged with other native DataStore objects.

In the case of connection type SAP HANA Smart Data Access, virtual tables are created for the administration tables when DataSources are activated for the read and write access to these tables in the native DataStore object. The virtual tables are displayed in the *Properties* view under *Extraction*.

Modeling aspects for DataSources with a native DataStore object with change log as source (delta extraction)

If the source object of the DataSource is a native DataStore object with change log, this has the following effects on the definition of the DataSource in terms of the delta extraction:

1. The delta process for the DataSource is set to *Additive or overwrite delta (delta queue) ser. by package*.
2. The delta process cannot be changed, and no properties can be maintained for generic deltas.
3. In addition to the columns in the active table of the native DataStore object, three further columns from the change log are automatically mapped to technical DataSource fields in the DataSource's field list:

Column	is mapped to	DataSource field
`technicalKey.activationId`		`TECHNICALKEY_ACTIVATIONID`
Column `technicalKey.recordNo`		DataSource field `TECHNICALKEY_RECORDNO`
Column `technicalAttributes.recordMode`		DataSource field `TECHNICALATTRIBUTES_RECORDMODE`

4. The names and data types of the three additional technical fields cannot be changed.

When activating the DataSource, two additional views are generated and displayed in the *Properties* view under *Extraction* - a view for the query access and the full extraction from the active table of the native DataStore

object, and a view for the delta extraction from the change log. The view for the query access maps the three additional fields to constants.

A data transfer process with a DataSource of this kind as source and extraction mode delta can trigger a process chain for streaming.

Runtime Behavior

If a data transfer process is executed, which has a DataSource as its source, which itself as a native DataStore object as its source, and the status of the DTP request thus changes to Active, a new load process is registered in the administration tables of the native DataStore object. In the case of full extraction or delta initialization, the view on the active table is automatically used as the source. The native DataStore object then blocks data administration operations such as rollback or activation of new data until the load process ends with status green or red. This makes it possible to transfer data by packet.

More Information

SAP HANA Data Warehousing Foundation - Native DataStore Object Administration Guide on the SAP Help Portal page for SAP HANA Data Warehousing Foundation 2.0 SPS 01 or higher, under https://help.sap.com/viewer/p/SAP_HANA_DATA_WAREHOUSING_FOUNDATION

4.14.1.2.3 Generic Delta for SAP HANA DataSources

The generic delta is a delta calculation mechanism for DataSources . It can be used for data transfer from SAP HANA - provided that the source object itself does not provide a delta process.

Using a delta-relevant field whose value rise monotonously over time, the system determines which data to transfer at runtime. For example, if a numerical pointer of this type of delta-relevant field exists, the system determines the data volume transferred in the delta mode by comparing the maximum value transferred with the last load with the amount of data that has since entered the system. Only the new data is transferred. An interval for the delta-relevant field is added to the request selections. Using safety intervals at the upper or lower limit of this interval, you can ensure that data records that are created during an extraction process but cannot be extracted (because they could not be saved for example) are looked at during the next extraction. The current upper stamp for the delta-relevant field is stored in a delta pointer table. The next delta uses the last delta stamp to find out which data has to be transferred.

For SAP HANA DataSources with connection type Local SAP HANA Database Schema with a table or a view as a source object, the generic delta is optimized if specific prerequisites are met in terms of deletion, streaming and timestamps as numeric pointers.

Procedure

In the DataSource editor, you have opened the *Extraction* tab page and chosen the delta process *Overwrite delta without deletions (BW local)* or *Pure additive delta (BW local)* for your DataSource:

- With the *Overwrite delta without deletions (BW local)* procedure, each record to be loaded returns the new status for all characteristics and key figures. The values in SAP BW/4HANA are overwritten. DataSources with this delta type can write data to DataStore objects and master data tables.
- With the *Pure additive delta (BW local)* procedure, the record to be loaded for cumulative key figures only returns the change to the respective key figure. The extracted data is added to SAP BW/4HANA. DataSources with this delta type can write data to DataStore objects.

To use the generic delta, complete the following steps:

1. Expand the *Delta Properties* area on the *Extraction* tab page.
2. Enter the delta-relevant field type in the *Generic Delta* field. The following field types are available: numeric pointers, time stamps as numeric pointers, date fields and time stamps (short and long).
   Depending on the delta-relevant field type, the system may also show other fields.
3. Enter the delta-relevant field in the *Data Field* field.
4. The following entries depend on the delta-relevant field type:
   Numeric pointer and numeric pointers (timestamp):
   In the *Lower Interval* field, enter the lower safety interval in the units of the numeric pointer field.

   > ⓘ Note
   >
   > For delta determination using a numeric pointer, it is not possible to determine an upper limit the delta, as the system does not know the highest value of the numeric pointer field when the request starts. It is therefore not possible to enter an upper safety interval for the numeric pointer.

   Time Stamp Long (UTC):
   Use *Time Stamp Long (UTC)* if the data is stored in Coordinated Universal Time (UTC).
   In the *Lower Interval* or *Upper Interval* field, enter a lower or upper safety interval in seconds.
   Date and Time Stamp Short:
   Use *Time Stamp Short* if the data is stored in a local (system) time zone.
   Only use *Date* if no other delta-relevant fields are available with a finer granularity, such as time stamps or numeric pointers.
   1. Enter the time zone.
   2. In the *Lower Interval* or *Upper Interval* field, enter a lower or upper safety interval - in days for date fields and in seconds for time stamp fields.

> ⓘ Note
>
> When using *Date*, note that it is always necessary to weigh up excluding the data that was changed today from the extraction, multiple extraction of data changes, and the risk of skipping data changes.
>
> For more information, see *Effects of the Safety Interval Settings for the Date as Delta-Relevant Field*.

Example of Determining Selection Intervals with Generic Delta

Safety interval upper limit

The delta-relevant field is a time stamp.

The last used upper limit is 12:00:00. The delta extraction commences as 12:30:00. The safety interval for the upper limit is 120 seconds. The selection interval for the delta request is then: 12:00:00 to 12:28:00. At the end of the extraction, the pointer is set to 12:28:00.

Safety interval lower limit

The delta-relevant field is a time stamp. After images are transferred. This means that the record in the SAP BW/4HANA system is overwritten with the status after the change, for example for master data. If duplicate data records are transferred, this does not have any effect in SAP BW/4HANA.

The last time stamp read is 12:28:00. The delta extraction commences as 13:00. The safety interval for the lower limit is 180 seconds. The selection interval for the delta request is then: 12:25:00 bis 13:00:00. At the end of the extraction, the pointer is set to 13:00:00.

Related Information

Optimizations for Connection Type Local SAP HANA Database Schema [page 483]
Effects of the Safety Interval Settings for the Date as Delta-Relevant Field [page 485]

4.14.1.2.3.1 Optimizations for Connection Type Local SAP HANA Database Schema

For SAP HANA DataSources with connection type *Local SAP HANA Database Schema* with a table or a view as a source object, the generic delta is optimized if specific prerequisites apply related to deletion, streaming and timestamps as numeric pointers.

Prerequisites

The prerequisite for optimizations is that the table or the view of the local SAP HANA schema is the target of an external real-time replication process (in other words, a process not managed by SAP BW/4HANA). In this case, “external” means that SAP BW/4HANA does not control the replication process by generating SAP HANA remote subscriptions. Instead, a different application controls the replication process, and the target table or target view of this replication process in the local SAP HANA schema is connected to SAP BW/4HANA via a SAP HANA DataSource. This type of application can either be a SAP HANA replication task or a flowgraph in the same SAP HANA, or an SLT replication server or another ETL process.

Delta with Deletions

If the above-mentioned prerequisites are met, you can use the delta process *Overwrite delta with deletions (delta queue)* in order to calculate the SDI_RECORD_MODE column from the value in the Change Type column.

> **Note**
>
> If the source object of the DataSource is filled using a SAP HANA replication task or a flowgraph, make sure that the load behavior is specified for the replication task or the flowgraph *Replication with logical delete* and that a *Change Type Column* has been specified in the replication task or flowgraph editor.

Proceed in SAP BW/4HANA as follows:

1. Go to the *Extraction* tab page and choose *Change* under *Delta with Deletions*.
   The *Change Settings for Delta with Deletions* wizard opens.
2. Select the field *Enable Delta with Deletions* and choose *Next*.
3. In the next wizard step, choose a column (from the source object) for the change type and choose *Next*.
   You can choose from a selection of `CHAR (1)` fields. The selected column is mapped to the DataSource field `SDI_CHANGE_TYPE`. No other DataSource fields can have the name of this one.
4. Please note the effects of changing the delta process on the data transfer processes and Open ODS views that use this DataSource, and choose *Finish*.
   The delta process of the DataSource changes to *Overwrite delta with deletions (delta queue)*.
5. Specify the properties for the generic delta in the *Delta Properties* area.
6. Continue with the further editing of the DataSource and specify the data type mapping for the DataSource fields.
7. Activate the DataSource.

When the DataSource is activated, two views are created on the target table or target view; one view for query access and full extraction, and one view for delta requests and delta initialization requests. DTP requests or Open ODS views automatically use the correct view at runtime.

The view for query access and full extraction automatically filters out deletions. The view for delta extraction calculates the record mode as an additional field `SDI_RECORD_MODE`. This field converts the change type into the corresponding values of the technical characteristic `0RECORDMODE`.

Streaming

A data transfer process, which extracts deltas from a table or view of the local SAP HANA schema, can trigger a streaming process chain, provided that specific prerequisites are met for the delta-relevant field. These prerequisites are listed in SAP Note 2372534.

Time stamp as numeric pointer

For source object columns with SAP HANA data type SECONDDATE or TIMESTAMP, you can choose *Numeric Pointer (Time Stamp)* as the type of delta-relevant field.

You can map SECONDDATE to ABAP data type `DEC (15,0)` or `NUMC (14)`, and map TIMESTAMP to `DEC (21,7)` or `NUMC (21)`. In this way, you can avoid incorrect time stamps being generated as the type of delta-relevant field in specific cases (a problem which can occur with *Numeric Pointer*).

4.14.1.2.3.2 Effects of the Safety Interval Settings for the Date as Delta-Relevant Field

If you specify *Date* as the type of the delta-relevant field, you need to consider a number of things when setting the lower and upper ranges of the safety interval.

1. Only choose *Date* if no other delta-relevant fields are available with finer granularity, such as time stamps or numeric pointers.
2. If you choose *Date*, there is always a trade-off between excluding the data changed, transferring data changes multiple time, and and the risk of missing out data changes.
    1. If you leave the lower safety interval (*Lower Interval (Days)* field) blank, the delta interval starts one day after the end date of the previous selection.
    2. If you enter **1** for the lower safety interval, the delta interval starts on the end date of the previous selection. This means that all records whose delta-relevant field contains the end date of the previous selection are read again.
    3. If you leave the upper safety interval (*Upper Interval (Days) field*) blank, one day is used as the default date. This ensures that data that is changed later in the day is not missed out.
    4. If you enter **-1** as the upper safety interval, the delta interval contains the current date. This means that data that is changed later on the same day is missed out if you leave the lower safety interval blank.
3. Here are some typical examples. Let us assume that today's date is 7th July, the end date of the previous selection was 4th July.
    1. If you leave the lower and upper safety interval blank, the delta selection starts on 5th July and ends on 6th July. This means that there are no duplicates and there is no risk of missing out data changes.
    2. If you enter **1** for the lower safety interval and **-1** for the upper security interval, the delta selection starts on 4th July and ends on 7th July. This means that data changes from 4th July and 7th July can be transferred twice.
    3. If you leave the lower safety interval blank and enter **-1** for the upper safety interval, the delta selection starts on 5th July and ends on 7th July. This means that data that is changed later on 7th July will be missed out.
4. Moreover, generic delta extraction can only be requested once per day if you leave the lower and upper safety interval blank or specify the same value for both. If you trigger delta extraction more than once per day, the requests will fail with error message RSDS 240. This message indicates that the lower bound for generic delta extraction is greater than the upper bound.

4.14.1.3 Creating a Big Data DataSource

Procedure

1. You are in the BW Modeling Tools. In your BW project in the *Data Sources* tree, open the context menu on the Big Data source system or the application component of the source system, and choose *New* > *DataSource*.
   The wizard for creating DataSources appears.
2. Make the required entries in the wizard.
   a. If necessary, specify an application component using *Browse*.
   b. To ensure that you can find your DataSource again without difficulty, we recommend setting the *Add to Favorites* flag.
   c. Specify the DataSource name, description, and type.
   d. Choose *Table/View* and use *Browse* to select the table or view in the source.
   e. Choose *Finish*.

   The DataSource editor appears.
3. On the *Overview* tab, you make general settings, if required.
4. On the *Extraction* tab, you make extractor-specific settings.
5. On the *Fields* tab, you maintain the properties of the DataSource fields.
   a. Specify the DataSource fields, which you want the system to use for package by package processing in the data transfer process when reading data from the source object. To do this, select the relevant field in the field list and select the *Use for Packaging and Partitioning* indicator in the field properties under *SDA-Specific Properties*. If you select properties for multiple fields, disjunctive selections are made across all fields selected.
6. Save and activate the DataSource.
7. Once you have created the DataSource, choose *Refresh* in the context menu of the source system or the application component.

Results

The DataSource is displayed in the *Data Sources* tree in the application component of the source system and is ready for use.

When you activate the DataSource, the system generates various objects such as the transfer program. In the *Properties* view, you can display the generated objects on the *Technical Attributes* tab page.

In the *Properties* view, you can display the objects generated for SAP HANA Smart Data Access on the *Extraction* tab page. For the query access, a virtual table and a view on the virtual table are created. The main task of this view is the type conversion of SAP HANA data types and ABAP data types.

In the *Properties* view, you can call a generated object by clicking the mouse.

4.14.1.4 Creating DataSources for ODP Source Systems

Before you can transfer data from an ODP source system, the metadata from the source in must be available in the SAP BW/4HANA system in the form of a DataSource. In addition to replicating the DataSource from a replicating ODP source system, you can also create a DataSource in the SAP BW/4HANA system manually using the DataSource Wizard.

Prerequisites

If there are no suitable application components to group the DataSources with, create an application component in the *Data Sources* tree for the ODP source system.

Procedure

1. You are in the BW Modeling tools. In your BW project in the *Data Sources* tree, open the context menu on the ODP source system or the application component of the source system, and choose *New* > *DataSource*.
   The wizard for creating DataSources appears.
2. Enter the required information the wizard pages and then choose *Finish*.
   The DataSource editor appears.
3. On the *Overview* tab, you make general settings, if required.
4. On the *Extraction* tab, you make extractor-specific settings.
5. On the *Fields* tab, you maintain the properties of the DataSource fields.
6. Save and activate the DataSource.

Result

The DataSource is displayed in the *Data Sources* tree in the application component of the source system and is ready for use.

You can now set a transformation between the DataSource and an InfoProvider (DataStore object) and load the data to the InfoProvider using a data transfer process. Alternatively, if the DataSource supports direct access, you can use it as a source in an Open ODS view.

In the SAP BW/4HANA system, you can monitor the data transfer with the monitor for data transfer processes. In the source, you can monitor the extraction and replication using the delta queues. The exception here is SAP ByDesign systems, as they do not use delta queues when implementing the ODP framework.

Related Information

Creating a Data Transfer Process [page 535]
Monitoring Delta Queues [page 178]
Executing Replication [page 468]

4.14.1.5 Creating DataSources for File Source Systems

Before you can transfer data from a file source system, the metadata (the file and field information) must be available in SAP BW/4HANA in the form of a DataSource.

Prerequisites

Note the following with regard to CSV files:

- Fields that are not filled in a CSV file are filled with a blank space if they are character fields or with a zero (0) if they are numerical fields.
- If separators are used inconsistently in a CSV file, the incorrect separator (which is not defined in the DataSource) is read as a character and both fields are merged into one field and can be shortened. Subsequent fields are then no longer in the correct order.
- A line break cannot be used as part of a value, even if the value is enclosed with an Escape sign.

Note the following with regard to CSV files and ASCII files:

- The conversion routines that are used determine whether you have to specify leading zeros.
- For dates, you usually use the format YYYYMMDD, without internal separators. Depending on the conversion routine used, you can also use other formats.

Notes on Loading

- When you load external data, you have the option of loading the data from any workstation into the SAP BW/4HANA system. Loading from a workstation is only possible in the dialog however. For performance reasons, you should store the data on an application server and load it from there into the SAP BW/4HANA system. This means that you can also load the data in the background and thus use it in process chains too.
- If you want to load a large amount of transaction data into the SAP BW/4HANA system from a flat file and you can specify the file type of the flat file, you should create the flat file as an ASCII file. From a performance point of view, loading data from an ASCII file is the most cost-effective method. Loading from a CSV file takes longer because the separator characters and escape characters then have to be sent and interpreted. In some circumstances, generating an ASCII file can involve more effort.
- When loading from a workstation, the data is loaded to a file in a single data package. There are various ways of avoiding this:
  - We recommend placing the file on the application server. From the application server, data can be read by package. Note however that Microsoft Excel files cannot be read from the application server. You should therefore convert your file into a CSV file.

- If possible, an alternative solution is to split your file into multiple smaller files.

Procedure

1. You are in the BW Modeling tools. Here, from your BW project, go the *Data Sources* tree for you source system and choose *New* > *DataSources* from the context menu.
   The wizard for creating DataSources appears.
2. Make the required entries in the wizard and then choose *Finish*. More information: Creating a DataSource Using the Wizard [page 472]
   The DataSource editor appears.
3. Make general settings for the DataSource on the *Overview* tab.
   1. [Optional:] Enter a description for the DataSource.
   2. If required, specify whether the DataSource builds an initial non-cumulative and has the ability to return duplicate data records in a request.
4. On the *Extraction* tab page, specify the extraction properties. More information: Editing Extraction Properties [page 489]
5. On the *Fields* tab page, define the field list and the properties of the fields. More information: Editing the Field List and Field Properties [page 494]
6. Check, save and activate the DataSource.
7. By choosing *Show Data Preview* in the editor toolbar or *Window* > *Show View* > *Others*, you can call the *Data Preview for DataSource* view. In the *Data Preview for DataSource* view, the system displays a preview of the active version of the DataSource. Here you can specify how many data records to display in the preview. For hierarchy DataSources, you can select which segment to the data is displayed for.
   This function allows you to check whether the data formats and data are correct.

Result

In the BW Modeling Tools, the DataSource is displayed in the *Data Sources* tree in the application component of the source system and is ready for use.

4.14.1.5.1 Editing Extraction Properties

On the *Extraction* tab, you define various properties for data extraction.

Procedure

1. Define the delta process for the DataSource.

You can use the generic delta. Using a delta-relevant field whose value rise monotonously over time, the system determines which data to transfer at runtime. More information: Using Generic BW Deltas [page 492].

2. Specify whether you want the DataSource to support direct access to data.
3. Select the adapter for the data transfer.
   For DataSources of type Master Data Attributes, Master Data Texts, and Transaction Data, you can select one of the following adapters:
   - Load Binary File from Local Workstation
   - Load Binary File from Application Server
   - Load Microsoft Excel File (XLS) from Local Workstation
   - Load Text-Type File from Local Workstation
   - Load Text-Type File from Application Server

   For DataSources of type Hierarchies, you can select one of the following adapters:
   - Load Text-Type File from Local Workstation
   - Load Text-Type File from Application Server

   Text-type files only contain characters that can be displayed and read as text. CSV and ASCII files are examples of text files. For CSV files, you have to specify a character that separates the individual field values. In the SAP BW/4HANA system, you have to specify this separator and an escape character which specifies this character as part of the value if required. You then have to use these characters in the file. ASCII files contain data in a specified length. The defined field length in the file must be the same as the length of the assigned field in the SAP BW/4HANA system.
   Binary files contain data in the form of bytes. A file of this type can contain any type of byte value, including bytes that cannot be displayed or read as text. In this case, the field values in the file in internal format must match the assigned fields in the SAP BW/4HANA system.
4. Choose *Browse* to select the path to the file that you want to load, or enter the name of the file directly, for example `C:/Daten/US/Kosten18.csv`.
   If you load text-type files or binary files, you can also create a routine to define the name of your file. If no routine has been created for the name of the file, the file name in the *File Name* field is interpreted directly.
5. Depending on the adapter and the file to be loaded, there are further settings that need to be made.
   - For binary files from the application server:
     Specify the character record settings for the data that you want to transfer: By default, the system code page is applied to the data to be transferred. If you want to define the character set settings differently, choose *Specify Codepage to Replace System Codepage* and make the required settings.
   - For text-type files:
     1. Under *Ignore Header Rows*, specify how many rows in your file are header rows and can therefore be ignored when the data is transferred.
     2. Specify the character record settings for the data that you want to transfer: By default, the system code page is applied to the data to be transferred. If you want to define the character set settings differently, choose *Specify Codepage to Replace System Codepage* and make the required settings.
     3. Under *Data Format*, specify which format the data in the file is in, in a fixed length or separated by separator characters:
        If you are loading data from an ASCII file, the data is requested with a fixed data record length. In this case, select *Fixed Length* or *Hierarchy Fixed Length*.
        If you are loading data from a CSV file, select *Separated by Separator (Example: CSV)* or *Hierarchy Separated by Separator* and set the data separator and the escape character:
        1. In the *Data Separator* field, specify the separator that your file uses to divide the fields.

2. If the separator is a part of the value, the file indicates this by enclosing the value in particular start and end characters. Enter these start and end characters in the *Escape Sign* field.
3. In a text editor (Notepad for example), check the separator and the escape character currently being used in the file. These depend on the country version of the file you are using.
4. If you select the *Hex* flag, you can specify the data separator and the escape character in hexadecimal format. When you enter a character for the data separator and the escape character, these are displayed as hexadecimal code after the entries have been checked. A two character entry for a data separator or an escape sign is always interpreted as a hexadecimal entry.

> **Note**
>
> - Example of how to use data separators and escape characters: You have chosen the ; character as the data separator. A field in your file has the value 12;45 however. If you have specified “ as the escape character, the value in your file must be in format “12;45“ so that 12;45 is loaded into BW. The complete value that you want to transfer has to be enclosed by the escape characters.
> - If the escape characters do not enclose the value but are used within the value, the system interprets the escape characters as a normal part of the value. If you have specified " as the escape character, the value 12"45 is transferred as 12"45 and 12"45" is transferred as 12"45".
> - Note that if you do not specify an escape character, the space character is interpreted as the escape character. We recommend using a different character as the escape character.

- For Microsoft Excel files:
  1. Under *Extract Sheet*, specify which of the file’s worksheets the data should be transferred from.
  2. Under *Ignore Header Rows*, specify how many rows in your file are header rows and can therefore be ignored when the data is transferred.

6. Under *Conversion Properties*, make the settings for the conversion language and the number format (thousand separator and character used to represent a decimal point) if necessary.
7. Make any further settings that are dependent on your selection if necessary.
8. For text or Microsoft Excel files, you can create a proposal for the field list of the DataSource based on the sample data in your file. To do this, choose *Derive Fields from File*. The wizard for updating DataSources appears.

> **Note**
>
> For hierarchies, this function is not available, since you use the hierarchy settings to define the structure of the hierarchy DataSource.

1. In the first step in the wizard, specify how many data records you want to use to generate the proposal for the field list.
2. If the file has header rows, and you have set a fixed length as the data format, you can choose whether the initial proposal for the separation of the fields is based on the header row or on the first data row.
3. Choose *Next*.
4. If a fixed length was set as the data format, you can specify the positions where the separation of fields occurs in the next step.
5. Choose *Next*.
6. In the wizard step *Confirm Update DataSource*, the system displays the proposed changes for the field list. Choose *Finish* in order to adjust the field list of the DataSource.

9. For hierarchies, define the structure of the DataSource by pressing *Change Hierarchy Settings*. The wizard for defining the hierarchy settings appears.
    1. In the first step, make the required settings. For this, you can specify the InfoObject that you are loading hierarchy data for as the target InfoObject. The hierarchy settings are then copied from the InfoObject.
    2. You can change the copied settings.
    3. Choose *Next*.
    4. In the wizard step *Confirm Changes*, the system displays the proposed changes for the field list. Choose *Finish* in order to adjust the field list of the DataSource.
10. For hierarchies, perform the following additional steps:
    1. Under *Maintain Hierarchy Header* in the DataSource editor toolbar, create a hierarchy header. You need this before loading, as the file does not fill the hierarchy header and header text segments. If the entire hierarchy is time-dependent, the Valid to and Valid from fields are input-ready.
    2. To display the proposed structure of the hierarchy file based on the hierarchy-specific settings, select *Show File Structure*.
    3. To show what a file with this structure will look like, select *Create Example File*. This file is a very simple hierarchy with just one root node, two subnodes and a leaf for each subnode. It does contain all properties defined for the hierarchy though, for example whether the hierarchy nodes are time-dependent or are intervals.

4.14.1.5.1.1 Using Generic BW Deltas

The generic BW delta is a delta calculation mechanism for DataSources from non-SAP source systems. It can be used to transfer data from files.

Using a delta-relevant field whose value rise monotonously over time, the system determines which data to transfer at runtime. If a delta-relevant field of this type exists (a time stamp, for example), the system determines the data volume transferred in the delta mode by comparing the maximum value transferred with the last load with the amount of data that has since entered the system. Only the new data is transferred. An interval for the delta-relevant field is added to the request selections. Using safety intervals at the upper or lower limit of this interval, you can ensure that data records that are created during an extraction process but cannot be extracted (because they could not be saved for example) are looked at during the next extraction. The current lower and upper stamps for the delta-relevant field are stored in a delta pointer table. The next delta uses the last delta stamp to find out which data has to be transferred.

Procedure

You have defined the structure of the DataSource in the DataSource editor and have selected the delta process *Overwrite delta without deletions (BW local)* or *Pure additive delta (BW local)* on the *Extraction* tab page:

- With the *Overwrite delta without deletions (BW local)* procedure, each record to be loaded returns the new status for all characteristics and key figures. The values in SAP BW/4HANA are overwritten. DataSources with this delta type can write data to DataStore objects and master data tables.
- With the *Pure additive delta (BW local)* procedure, the record to be loaded for cumulative key figures only returns the change to the respective key figure. The extracted data is added to SAP BW/4HANA. DataSources with this delta type can write data to DataStore objects.

You can now expand the *Delta Properties* on the *Extraction* tab page.

To be able to use the generic BW delta, complete the following steps:

1. Select the type of the delta-relevant field under *Generic Delta*. The following field types are offered: Numeric pointers, date fields and time stamps (short and long)..
   Depending on the delta-relevant field type, the system may also show other fields.
2. Under *Delta Field*, select the delta-relevant field. You can select from the fields that are valid for the selected type of data field. For dates, only fields with the data type DATS are displayed, while for time stamps only fields of data type DEC or NUMC are displayed. For numeric pointers, all fields are displayed.
3. The following entries depend on the delta-relevant field type:
   Numeric pointer and numeric pointers (timestamp):
   In the *Lower Interval (Units)* field, enter the lower safety interval in the units of the numeric pointer field.

   > ⓘ Note
   >
   > For delta determination using a numeric pointer, it is not possible to determine an upper limit the delta, as the system does not know the highest value of the numeric pointer field when the request starts. It is therefore not possible to enter an upper safety interval for the numeric pointer.

   Time Stamp Long (UTC):
   Use *Time Stamp Long (UTC)* if the data is stored in Coordinated Universal Time (UTC).
   In the *Lower Interval (Seconds)* or *Upper Interval (Seconds)* field, enter a lower or upper safety interval in seconds.
   Date and Time Stamp Short:
   Use *Time Stamp Short* if the data is stored in a local (system) time zone.
   1. Enter the time zone.
   2. In the *Lower Interval (Seconds)* or *Upper Interval (Seconds)* field, enter a lower or upper safety interval - in days for date fields and in seconds for time stamp fields.

Example of Determining Selection Intervals with Generic Delta

Safety interval upper limit

The delta-relevant field is a time stamp.

The last time stamp read is 12:00:00. The delta extraction commences as 12:30:00. The safety interval for the upper limit is 120 seconds. The selection interval for the delta request is then: 12:00:00 to 12:28:00. At the end of the extraction, the pointer is set to 12:28:00.

Safety interval lower limit

The delta-relevant field is a time stamp. After images are transferred. This means that the record in the SAP BW/4HANA system is overwritten with the status after the change, for example for master data. If duplicate data records are transferred, this does not have any effect in SAP BW/4HANA.

The last time stamp read is 12:28:00. The delta extraction commences as 13:00. The safety interval for the lower limit is 180 seconds. The selection interval for the delta request is then: 12:25:00 to 13:00:00. At the end of the extraction, the pointer is set to 13:00:00.

4.14.1.5.2 Editing the Field List and Field Properties

On the *Fields* tab, you define the field list of the DataSource and the properties of the fields.

Edit Field List

1. If the field list is generated from the file with a proposal, the system displays this proposal under *Fields*. You can change the field list here or create one if there is no proposal.
2. You can locate changes between the file and the field by choosing *Update DataSource*. If differences are found, the system displays a change proposal. You can then decide whether or not to apply the proposed changes in the field list. More information: Metadata Comparison and Updating DataSources [page 497]
3. To add fields to the field list manually, choose *Add Field* or *Add InfoObjects*.

   > **Note**
   >
   > This is not possible for hierarchy DataSources, since the structure of the DataSource is defined using the hierarchy settings on the *Extraction* tab.

   1. Instead of generating a proposal for the field list, you can enter InfoObjects to define the fields for the DataSource. To do this, choose *Add InfoObjects*. The system displays a dialog box where you can add one or more InfoObjects from the list to the selection pane. You can also copy a list of InfoObject names to the clipboard and add this to the selection.
   2. Choose *OK* to add all corresponding fields to the field list of the DataSource for all InfoObjects from the selection area. This field name and description are derived from the InfoObject, and the technical properties of the InfoObject are added to the DataSource field.
      In the field list, the InfoObject that the field attributes were copied from is displayed in the *InfoObject Template* column.

   > **Note**
   >
   > Entering InfoObjects here does not equate to assigning them to DataSource fields. This assignment is made in the transformation. When you define the transformation, the system proposes the InfoObjects you entered here as InfoObjects that you might want to assign to a field.

4. Choose *Up* or *Down* in order to move selected fields up or down in the field list.
5. Choose *Remove* in order remove selected fields from the field list.
6. Under *Transfer*, specify the decision-relevant DataSource fields that you want to be available for extraction and transferred to the SAP BW/4HANA system.
7. In the context menu for a field, specify the field type, in other words whether the data is language-dependent or time-dependent.

Maintain Field Properties

For manually created fields, you re-define the properties. For fields that are based on a proposal, you can change the properties that were derived or taken over from the proposal.

> **Note**
>
> For fields of hierarchy segments, not all field properties can be edited.

1. Select the field whose properties you want to edit.
2. Under *General*, enter a field name and a description (optional).
3. Under *Data Type* , enter the data type and, depending on the selected data type, the length of the field or the number of digits and number of decimal places.
4. Under *InfoObject template*, you can also add the technical properties of an InfoObject to the field at field level.

   > **Note**
   >
   > Entering an InfoObject here does not equate to assigning it the DataSource field. This assignment is made in the transformation. When you define the transformation, the system proposes the InfoObject you entered here as an InfoObject that you might want to assign to a field.

5. Under *Technical Properties*, make the following entries:
   1. For an unit or currency field: Under *Unit/Currency Field*, select the referenced currency or unit field or, if the field has a fixed currency or unit, specify the currency or unit directly under *Fixed Unit/Currency*. If you specify the referenced currency or unit field, this field must be contained in the DataSource structure and be of type CUKY (currency) or UNIT (unit).
      Specifying a reference field or a fixed currency is only necessary if the amount field is delivered in external format. For quantities (data type QUAN), the specification is purely for information purposes and is not checked at runtime.
   2. Under *Key Field*, specify whether the field is a key field of the DataSource.
   3. For a character-type field: Select *Case-Sensitive* if lower case letters are supported in characteristic values.
      The property can be edited for the following data types: CHAR, SSTR, STRG, LANG, CUKY, and UNIT. The system checks the data type of the DataSource field, not the data type of the source field. If the data type of a type that does not support the property is changed into a type that does support the property, the *Case-Sensitive* property is selected by default. If the data type is changed in such a way that it does not support the *Case-Sensitive* property, the selection is removed.
6. Under *Conversion Properties*, make the following entries:
   1. Specify whether the source returns the data in internal format or external format.
      Source data that is transferred as text can differ from the internal, SAP character representation, that is, the internal format. Choose one of the following options:
      - *Internal*: Only choose this option if you are sure that the data source returns the data in internal format for this field. If this is the case, no checks and conversions are performed. If the source does not return an internal format, there could be data inconsistencies or errors in the loading process. Example: A date field is always returned with data in format yyyymmdd.
      - *External*: Select this option if you know that the data is not in internal format and requires conversion to internal format before being passed on to the transformation. Example: The data for a date field has the format dd.mm.yyyy. It should be converted to the internal format yyyymmdd.
      - *Check* Choose this option if the data from the souce should be be in internal format, but needs to be checked before being passed on to the transformation. Choose this option to ensure data conformity. Example: The source should return format yyyymmdd for a date field, but this is not the case.
   2. If necessary, under *Conversion Routine*, specify a conversion routine that converts data from external format to internal format.

3. If you choose external format, make sure under *External Length* that the output length of the field (external length) is correct. Change the entries if necessary.

7. Under *Selection Properties*, make the following entries:
   1. Under *Selection*, specify whether defining selections is allowed for the field when scheduling a data request, or whether selections for the field are required in order to be allow extraction of data. The selectable fields can be found in the data transfer process for the filter. Data for this type of field is then transferred in accordance with the selections (filter) specified in the data request.
   2. Under *Selection Options*, define which selection options (such as EQ, BT) you want to be available for selection.

Mass Maintenance of Field Properties

If you select multiple fields in the field list, you can use mass maintenance to change the properties of the selected fields. If the selected fields have different values for a particular property, this is indicated by the expression <multiple values> or the corresponding flagging of a checkbox. A change made to a property applies to all selected fields. If a property cannot be changed for one of the selected fields, the property cannot be changed in mass maintenance. If the selected fields have different data types, no information is displayed for the length, the number of digits, and the number of decimal places.

4.14.2 Editing DataSources

You can call the DataSource maintenance transaction from the context menu for a DataSource.

Procedure

You are in the *Project Explorer* view in the BW Modeling tools.

Call the context menu for the DataSource in the *Data Sources* tree and choose *Open*.

Results

Depending on the source system type and the source field, you can change certain properties of the DataSource and the DataSource fields.

For SAP HANA DataSources and ODP DataSources, you can exchange the source object in the editor. If you want to exchange the source object of the DataSource, press the *Exchange* button and then select the new source object. The *Confirm Update DataSource* opens. This takes an initial proposal from the back end to update the DataSource, and displays this in comparison with the current DataSource version. When editing SAP HANA DataSources, note that native DataStore objects can only be exchanged with other native DataStore objects, and tables/views can only be exchanged with other tables/views.

4.14.3 Metadata Comparison and Updating DataSources

In the DataSource Editor in the BW Modeling tools, you can perform a metadata comparison between the source object and the DataSource in SAP BW/4HANA and update the DataSource in SAP BW/4HANA.

You are in the DataSource Editor.

1. In the editor toolbar, choose *Update*.
   The *Confirm Update DataSource* wizard appears. For the various areas header data, extraction, segment properties and fields, the wizard indicates whether there are differences to the source. The wizard displays a proposal from the back end based on the metadata of the current object that the DataSource is based on. All relevant properties of the current M version of the DataSource are compared with the proposal.
2. You can use the *Only Changed* flag to specify that only differences detected in compared DatsSource properties are displayed instead of all DataSource properties.
3. In the *Take Proposal* column, the system indicates whether the value of a proposal should overwrite the current value in the DataSource.
   For all header data, segment properties and extraction properties, this is fixed and cannot be changed. The same applies for new or deleted fields. For differences in the field properties, you can decide whether to accept the proposal at field level by making a selection in the column. For properties that cannot be maintained by the user and are set by the source, the selection for whether a proposal should be overwritten is set by the system.
   By pressing *Select All* and *Deselect All*, you can specify whether or not the proposal should be accepted for all fields at once.
4. To update the DataSource in accordance with your selection, choose *Finish*.
   When activating the DataSource, note that the transformation might become inactivated and require adjustment.

4.15 Creating Transformations

The transformation process allows you to consolidate, clean up, and integrate data.

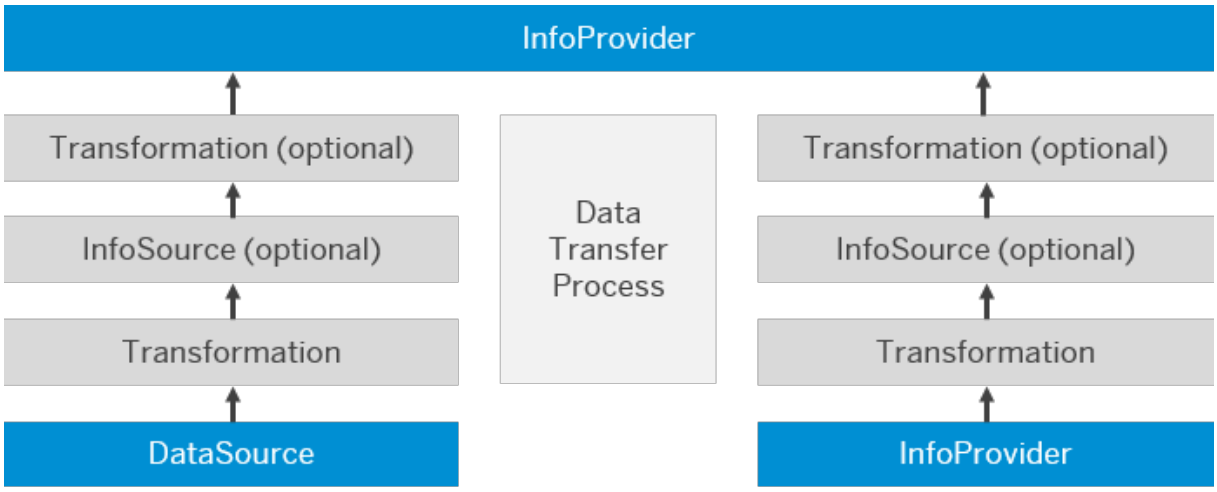
Context

You can semantically integrate data from various sources.

You create a transformation between a source and a target. When data is loaded from a BW object or virtual object to a target BW object, the data passes through a transformation. A transformation converts the fields of the source into the format of the target.

A transformation consists of at least one transformation rule. Various rule types, transformation types, and routine types are available. These allow you to create very simple and highly complex transformations.

The following figure illustrates how the transformation is integrated in the data flow:



Procedure

1. You are in the BW modeling tools, in the editing screen for your data flow. You have already created connections, and your objects are active. In the context menu of a data flow object, choose *Transformations* > *Create Transformation*. The wizard for creating transformations opens. More information: Creating a Transformation Using the Wizard [page 501]
2. The *General* tab opens.
3. With *Allow Currency and Unit Conversion*, you can enable currency translation and unit conversion for InfoSources and DataStore objects. As a prerequisite for this, all currencies and units of measure must be contained in the key of the target. For DataStore objects, translation/conversion of currencies and units is only supported with InfoObjects.
4. With *Check Units for Consistency*, you can activate the consistency check for units. This allows you to prevent records from being updated with initial units of measure for example.
5. In *Runtime Properties*, you can see whether the transformation is being processed in SAP HANA or in ABAP runtime. The default value is SAP HANA You can switch this value to process the transformation in ABAP runtime. For the switch, the system checks that the rules and any existing routines are still supported.

   Note:

   - The differences between the two terms: Differences Between ABAP Runtime and SAP HANA Runtime [page 500]
   - The supported objects: Transformation in the SAP HANA Database [page 502]

> ### Note
>
> In the case of an ODP source system, we recommend using the options provided in SAP Note *ODP Data Sources – Parallelize Filling of DTIS via Prefetch*. This can improve the performance of the overall execution if you want to extract a large number of data records from the DataSource and use certain functions such as SAP HANA runtime, error handling, grouping in the extraction in the data transfer processes. If you **do not** use these functions in the data transfer processes when extracting from the ODP DataSource - and use an InfoProvider in your data flow instead - you have the option of removing this intermediate level.

6. In *Global Routines* you can create routines for your transformation. Depending on the processing setting in *Runtime Properties* (SAP HANAor in ABAP runtime), you can create routines in SAP HANA SQLScript or ABAP routines. End routines can be created in SAP HANAruntime as a SAP HANA SAPScript or as an ABAP routine. In ABAP runtime, only ABAP routines can be created.

   More information: Routines in Transformations [page 511]

7. In *Extraction Grouped By*, you can select source fields for the semantic grouping. Data records with the same values in the selected fields form a logical key and are processed together in a package.

   More information: Semantic Grouping [page 189]

8. In *Additional Rule Groups* by choosing *Add* you can create additional rule groups.

   More information: Rule Group [page 188], Create Rule Group [page 501]

9. Choose the *Rules* tab.

   The system proposes a transformation. The proposal is generated on the basis of a comparison with existing transformations. The system also checks for Basis InfoObjects that can be assigned (UNIT to 0UNIT for example) or whether the DataSource can be assigned to InfoObjects while editing the DataSource.

   You can accept the proposed transformation or modify it. On the left side of the screen, you see the source; on the right the target.

   > ### Note
   >
   > For InfoProviders with non-cumulative key figures, you cannot change the transformation proposed by the system. These transformation rules fill the time reference characteristic of the InfoProvider. All other time characteristics are automatically derived from the time-reference characteristic.

10. With *Filter Pattern* you can filter the fields.
11. You can use the mouse to drag new connecting arrows. You can change or delete existing connecting arrows in the *Properties View*.
12. You can edit the rule details. More information: Editing Rule Details [page 503]
13. You can copy a rule by choosing *Copy Rule* in the context menu of a target field, and apply this rule in another target field by choosing *Paste Rule* from the context menu of this field. You can then change this rule.
14. The *Technical Rules* tab displays the rules of technical fields.
15. In *Show in History View* you get an overview of the versions of the transformation.
16. Activate the transformation.

Results

The transformation is executed with the corresponding data transfer process when the data is loaded.

To check whether the transofrmation provides the required results, you can simulate it to begin with. To do this, execute the simulation of the DTP request. The transformation is also simulated when the data update is simulated.

More information: Simulating and Debugging DTP Requests [page 546].

4.15.1 Differences Between ABAP Runtime and SAP HANA Runtime

The transformations can be processed in ABAP or in SAP HANA. You make this setting for a transformation explicitly on the *General* tab.

SAP HANA runtime provides better performance than ABAP runtime, and mass data can be processed very quickly. SAP HANA runtime is more complex to work with however.

There are essential differences between the two runtime types, which can lead to different results. When using SAP HANA runtime, you should therefore carefully validate the results against expected results, especially if existing ABAP transformations are to be replaced.

The main differences between the runtimes are described below:

Type Conversion

In ABAP runtime, types supported in ABAP are converted automatically, so a value of a character field can easily be written to an integer field if the field contains numbers at runtime. In SAP HANA runtime, a runtime error will occur if you do not explicitly specify the type conversion in the formula or in the SAP HANA script.

See example of a formula: Creating a Formula [page 507]

In all more complex transformations, you have to check that the result meets your expectations. As described above, different behavior and different results might occur.

Rounding of Values

Unlike with ABAP runtime, the results of calculations when working with SAP HANA are truncated according to the decimal places, and are not rounded.

> **Example**
>
> You have an *amount* key figure of type *DEC* (decimal 17) with three decimal places and the value 2.
>
> The formula AMOUNT / 3 calculates the value 0.667 (rounded up to the last decimal place) in ABAP runtime, but 0.666 in SAP HANA runtime.

If you use a formula, you can assure the rounding by additional conversions and an explicit rounding:

TO_DECIMAL(ROUND(TO_DOUBLE(AMOUNT) / TO_DOUBLE(/BIC/SZRDEC02); 3; ''); 17; 3)

Error Handling

You can use error handling for data records with errors in the DTP. More information: Specifying How Data Records Are Handled [page 541] .

4.15.2 Creating a Transformation Using the Wizard

Whenever you create a transformation, you are guided through the process by a wizard.

Context

Procedure

1. 1. You are in the BW Modeling tools. In the context menu for your BW project, choose *New* > *Transformation* .
2. Select the source object in *Source*.
3. You can select an object from which the transformation rules are to be copied.
4. When you have finished making your settings, choose *Finish*. The editor appears.

4.15.3 Create Rule Group

Context

Besides the standard rule group, you can create additional rule groups.

Procedure

1. You are on the *General* tab, editing your transformation. Under *Additional Rule Groups* enter a name for the group and choose *Add*. A new tab appears for the group.
2. The rules of the standard group are displayed as dashed lines on this tab. If you want to change one of these rules, in the *Properties View* you can choose the function *Copy Original Rule*. This rule is then displayed with a solid line, and can be changed.

3. With *Reset to Original* in the *Properties View*, you can reset the rule back to the rule for the standard group.

4.15.4 Transformation in the SAP HANA Database

If possible, transformations are processed in the SAP HANA database.

Processing in SAP HANAgenerally improves perfomance. By default, the transformation is therefore processed in SAP HANA You can switch this value to process the transformation in ABAP runtime. For the switch, the system checks that the rules and any existing routines are still supported.

For InfoObjects and data mart DataStore objects, there is combined SAP HANAand ABAP processing. Extraction and transformation are performed using SAP HANA and SID handling is executed and written using ABAP. This does not improve performance for standard transformations using 1:1 assignments. If you have selected *Read Master Data* or *Read from DataStore object* as the rule type, this leads to improved performance.

> **Note**
>
> As far as ODP source systems are concerned, there are cases where we recommend using ABAP runtime since the data must first be persisted and the extraction then takes place serially with ABAP. These cases do not meet the prerequisites for parallelized extraction set out in SAP Note 3232927.

Routines are created as SAP HANAscript or ABAP routines depending on the selected runtime.

More Information:

- SAP HANA SQLScript Reference
- Transformation in the SAP HANA Database [page 502]
- ABAP Routines in Transformations [page 511]

If you need special transformations that are not possible with the standard transformation rules, you can create these as expert routines as SAP HANAscript in SAP HANA. This function is intended for experts only. You need to be registered as a developer, in order to use this function.

> **Note**
>
> You have to model date calculations (such as sy-date + 1) with formula ADD_DAYS_TO_DATE, both for internal and external display.

Unsupported Objects

The following objects are not supported for executing transformations in SAP HANA:

- Objects with hierarchies
- InfoSources with the property that the data is to be aggregated
- Customer-defined functions in the formula editor
- Implicit type conversion in formulas
- To read data from DataStore objects, the entire key must be provided.

- Nearline connection

For more information, see SAP Note 2329819 .

4.15.5 Editing Rule Details

You can modify the generated rule details according to your needs.

Procedure

1. You are in the BW modeling tools, editing your transformation on the *Rules* tab.
2. Select a field/InfoObject whose rule you want to edit by double-clicking on the line representing the rule, or by using the context menu of the source or target field. The rule details are displayed in the *Properties View*.
3. With *Type* you can change the rule type.

   More information: Rule Type [page 504]
4. If you selected *Lookup* as the type, you have to assign an InfoObject or a DataStore object from which data is to be read from.

   Using *Map Fields of Lookup Object* you can display the assignment of the fields and assign missing fields. For DataStore objects, all key fields can be assigned to source fields.
5. You can activate the conversion routine, if it has been created. This is deactivated in the default setting, as the system assumes that the DataSource provides the internal format. The data should generally be in an internal format. This means that the conversion exit should be executed in the DataSource if possible. Only conversion routines ALPHA, GJAHR, NUMCV, PERI6 and PERI7 are supported.

   More information:
6. Under *Run Master Data check for InfoObject*, for individual characteristics you can activate the check for referential integrity. The check for referential integrity determines the validity of a characteristic's value before it is updated. The system checks if the master data table (attribute table) or DataStore object specified in the InfoObject maintenance for this characteristic contains an entry for this characteristic. If no entries are found, an error message is displayed. If a characteristic does not contain any attributes, the check is not offered.
7. With key figures, you can select the aggregation type. More Information: Aggregation Type [page 508]
8. For key figures, you can specify a transformation type. If you have set the indicator *Allow Currency and Unit Conversion* on the *General* tab, you can define a currency translation or quantity conversion.

   More information:

   Currency Translation During Transformation [page 509]

   Quantity Conversion During Transformation [page 510].

4.15.5.1 Rule Type

The rule type defines whether and how a characteristic/key figure or a data field/key field is updated into the target.

Direct Assignment:

The field is filled directly from the chosen source InfoObject. If the system does not propose a source InfoObject, you can assign a source InfoObject of the same type (amount, number, integer, quantity, float, time) or create a routine.

If you assign a source InfoObject to a target InfoObject that has the same type but a different currency, you have to translate the source currency to the target currency using a currency translation, or transfer the currency from the source.

If you assign a source InfoObject to a target InfoObject that has the same type but a different unit of measure, you have to convert the source unit of measure into the target unit of measure using a unit of measure conversion, or transfer the unit from the source.

Routine:

The field is filled by a transformation routine that you have written. Depending on the runtime selected, ypu have to implement the routine in ABAP or SQLScript SAP HANA

> ⓘ Note
>
> For DataStore objects and InfoObjects: You cannot use the return code in the routine for data fields that are updated by being overwritten. If you do not want to update specific records, you can delete these from the start routine.

A routine can have any number of fields as input fields and one field (data or key field) as the output parameter. Using the input parameters, and ABAP code or SAP HANASQLScript (depending on the selected runtime), you specify the value of the target field.

If you fill the target key figure from a transformation routine, currency translation has to be performed using the transformation routine. This means that no automatic calculation can be performed.

More Information:

- ABAP Routines in Transformations [page 511]
- SAP HANA SQLScript Routines in the Transformation [page 533]

Formula:

The InfoObject is updated with a value determined using a formula.

More information: Creating a Formula [page 507]

Read:

You can read data from InfoObjects or DataStore objects.

Read from InfoObjects:

The InfoObject is updated by reading the master data table of a characteristic from the source with a key and a value and contains the InfoObject with the same name as an attribute. The attributes and their values are read from the key and are then returned.

> ### Example
>
> The Financial Management Area characteristic is included in the target but is not a characteristic in the source. There is a characteristic (cost center, for example) in the source however. This has the Financial Management Area characteristic as an attribute. You can read the Financial Management Area attribute from the master data table and use it to fill the characteristic in target.

> ### Note
>
> It is not possible to read additional attributes for the attributes. You have to use routines for this.
>
> If you have changed master data, you have to execute the change run. This is because the active version is read when the master data is read.. If this cannot be found, the system raises an error.
>
> In transformations with an open hub destination, it is not possible to read from master data.

If the attribute is time-dependent, you also have to define the read time: On the current date (sy-date), at the beginning or end of a period (determined by the time characteristic in the InfoSource) or on a constant date that you enter directly. Sy-date is used in the default setting.

Read from DataStore Object (advanced):

To read from a DataStore object (advanced), the system first provides you with a proposal for how the fields can be assigned. The key fields must be assigned to the source fields of the transformation. When assigning the fields, the names do not need to be identical.

Constant:

The field is filled directly with the value entered.

InfoObject 0RECORDMODE is an exception to this. The system chooses *Constant* for this, but no value is required. In this case, the constant is required for delta administration (after images) with DataStore objects or InfoObjects as InfoProviders. Apart from this, no records are deleted. If your DataSource return a suitable field for 0RECORDMODE, you can assign this directly instead.

Time Update:

When performing a time update, automatic time conversion and time distribution are provided.

Direct update: The system provides an automatic time conversion.

Time Conversion:

You can update source time characteristics to target time characteristics using automatic time conversion. This function is not available for DataStore objects, as time characteristics are treated as normal data fields. The system only offers time characteristics that have an automatic time conversion routine.

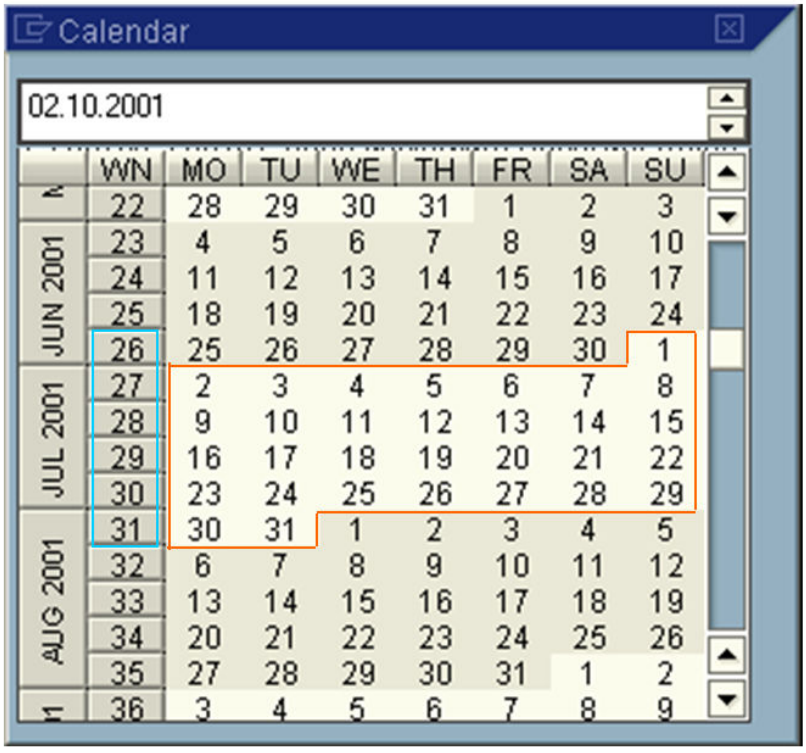
Time Distribution:

You can update time characteristics with time broadcasting. All the key figures that can be added are split into correspondingly smaller units of time. If the source contains a time characteristic (such as 0CALMONTH) that is not as precise as a time characteristic of the target (such as 0CALWEEK), you can combine these characteristics in the rule. The system then performs time broadcasting in the transformation.

Example

You might for example divide calendar month 07.2001 into weeks 26.2001, 27.2001, 28.2001, 29.2001, 30.2001 and 31.2001. Every key figure that can be added receives 1/31 of the original value for week 26.2001, 7/31 for each of weeks 27,28,29 and 30, and exactly 2/31 of it for week 31.

The example is clearer if you compare it with the calendar below:



Time broadcasting always applies to all key figures.

Initial:

Initial is the default rule type for key fields (and only possible for these). The correct type for the initial entry is written to the field.

Unit of Measure Conversion and Currency Translation:

You can convert data records into the unit of measure or currency of the target transformation.

For more information, see

- Currency Translation During Transformation [page 509]
- Quantity Conversion During Transformation [page 510]

0RECORDMODE Calculation for ODP

If the source of your transformation is a DataSource taht supplied with data via an Operdational Data Provider (ODP), and the target is a DataStore object or InfoObject, you need rule type *0RECORDMODE Calculation for ODP* for the ODQ_CHANGEMODE and ODQ_ENTITYCNTR source fields. This rule type can also be used to calculate the 0RECORDMODE field.

If you load deltas from an ODP that does not return just one image type (after images, delete images and new images for example), this change behaviour is provided by the ODQ_CHANGEMODE and ODQ_ENTITYCNTR fields. The 0RECORDMODE field has to be calculated from these fields for use in the BW system.

4.15.5.2 Creating a Formula

You can create a formula as a rule type.

Context

Various predefined functions are available to do this.

The InfoObject is updated with a value calculated by a formula.

> ⓘ Note
>
> In the SAP HANA runtime no implicit type conversion is supported for runtime/ modelling time.
>
> Example of a formula with differently typed formula elements (HX_INF6 of type C, 0 typed with I):
>
> Variant A: IF (HX_INF6 > 0, 10, 20)

Variant A can be executed in ABAP runtime because ABAP runtime ensures that the value of HX_INF6 is converted implicitly from type C to type I before the actual comparison HX_INF6 > 0 is executed. This variant cannot be executed in the SAP HANA runtime.

Variant B: IF(TO_INT(HX_INF6) > 0, 10, 20)

Variant B can be executed in SAP HANA runtime because the casting function TO_INT explicitly ensures that the value of HX_INF6 is converted from type C to type I before the actual comparison HX_INF6 > 0 is executed. This variant cannot be executed in ABAP runtime because the casting function TO_INT is not available in the ABAP stack.

Procedure

1. You are in the BW modeling tools, processing your transformation. You have selected *Formula* as the rule type.
2. In the *Properties View* next to the input field, choose the *Edit* formula. The formula editor appears.
3. You can define your formula in the *Formula* field. To simplify the task of formula definition, you can select the required operands and operators for the formula from *Formula Groups*. The system then inserts these into the formula definition with the correct syntax. You can use the numeric keypad on the right to enter basic arithmetic operations.

   With each step, the system checks the syntactic correctness of the formula. If the formula is incomplete or syntactically incorrect, the system underlines the error in red.

   > ⓘ Note
   >
   > Further information about the formula functions is provided under Formula Calculation [page 583].

4. Choose *Format Formula* to format your formula. In this step for example, the system removes any unneeded spaces and inserts any line breaks required.
5. Confirm your formula definition by pressing *OK*. The formula is displayed in the *Formula* field.

4.15.5.3 Aggregation Type

You use the aggregation type to control how a key figure or data field is updated to the InfoProvider.

For InfoObjects:

Only the *Overwrite* option is available. With this option, new values are updated to the InfoObject.

For DataStore Objects (advanced):

Depending on the properties of the DataStore object (advanced) in question, you are offered the option *Summation*, *Maximum*, *Minimum* or *Overwrite*. When you choose one of these options, new values are updated to the DataStore object.

If you have selected the modeling properties *Activate Data* and *All Characteristics are Key, Reporting on Union of Inbound and Active Table*, you can choose from the following aggregation types: *Summation*, *Maximum* and *Minimum*.

If you have selected the modeling properties *Activate Data* and *Write Change Log*, you can choose from the following aggregation types: *Summation*, *Maximum*, *Minimum* and *Overwrite*.

If you have not selected any modeling properties, you are not offered any aggregation types at all.

4.15.5.4 Currency Translation During Transformation

Currency translation during transformation enables you to convert data records from the source currency into a currency of the target of the transformation.

Currency translation during transformation is performed using previously defined translation types. More information: .

You can also create a currency translation for fields that are not in the key. If the fields are not in the key, there should be no more than one currency field in the target however.

Currency translation types that contain variables cannot be used in the transformation.

This can also be done using user-defined subprograms (formulas and routines).

Depending on the type of currency, there are two different types of key figures:

1. Key figures with a fixed currency (for example, DEM, USD)
   With a fixed currency, the currency is fixed with the amount. As the key figure refers specifically to the currency (for example DM, US$), this does not have to be entered again in the data record.
2. Key figures with a variable currency (foreign currency for example).
   With a variable currency, the currency is determined by an InfoObject, (ODOC_CURRCY) for example.

More information: Creating Key Figures [page 265]

Features

When transforming key figures, you have two options:

1. You can assign a source key figure of the same type to the target key figure (sales revenue instead of sales quantity revenue, for example).
   - If both key figures have the same currency, no currency translation is performed.
   - If the currencies are different, translation can be performed either using a currency translation type or by simply assigning a currency.
     The following table provides an overview of possible combinations where the currency is not the same in the source and target key figures:

Source Key Figure Currency	Target Key Figure Currency	Currency Translation (CT) Activated
Fixed	Variable	By default CT is deactivated, but it can be activated.

Source Key Figure Currency	Target Key Figure Currency	Currency Translation (CT) Activated
Fixed	Fixed	CT
Variable	Fixed	CT
Variable	Variable	CT or assignment possible

2. Translating the source currency: For fixed currencies you can define a field or InfoObject as a variable currency and vice versa.
3. If there is no corresponding source key figure of the same type, you can fill the key figure of the target using a routine.
   - If the currency of the target key figure is fixed, no currency translation is performed. This means that you have to perform the translation in a routine if required.
   - If the currency of the target key figure is variable, you also have to assign a variable source currency to the routine. You can use input help to select a currency from the variable currencies that exist for the target. You have two options:
     - You can assign the selected variable currency.
     - You can select a currency translation type and a currency to translate into ('to' currency). The target currency can be defined in the UI. This also applies for formulas.

4.15.5.5 Quantity Conversion During Transformation

Use

Quantity conversion during transformation enables you to convert data records from the source unit of measure into a unit of measure in the target of the transformation.

Quantity conversion during transformation is generally performed using previously defined conversion types. More information: .

This can also be done using user-defined subprograms (formulas and routines).

Depending on the type of unit of measure, there are two different types of key figures:

1. Key figures with a fixed unit of measure
   With a fixed unit of measure, the unit is fixed for the key figure. The key figure refers specifically to the unit of measure. meaning that the unit does not have to be entered again in the data record.
2. Key figures with a variable unit of measure
   A variable unit of measure references an InfoObject.

More information: .

Features

When transforming key figures, you have the following options:

1. You can assign a source key figure of the same type to the target key figure.
   - If the units of measure of both key figures are the same, no quantity conversion can take place.
   - If the units of measure are different, a conversion can take place either using a quantity conversion type or by simply assigning a unit of measure.
2. If there is no corresponding source key figure of the same type, you have to fill the key figure of the target using a routine.
   - If the unit of measure of the target key figure is fixed, no quantity conversion is performed. This means that you have to perform the conversion in a routine if necessary.
   - If the unit of measure of the target key figure is variable, you also have to assign a variable source unit of measure to the routine. You can use input help to select a unit of measure from the variable units of measure for the target. You have two options:
     - You select a variable unit of measure and assign it.
     - You select a quantity conversion type and a unit of measure that you want to convert to.

Conversion Using a Quantity Conversion Type

If you have chosen an InfoObject for determining the target unit of measure in the quantity conversion type, you must heed the following when maintaining the transformation rules:

The InfoObject for determining the target unit of measure must be contained in both the source and target systems and must be filled using a rule.

More information: .

Routines for Quantity Conversions

If you want to convert units of measure during the transformation but quantity conversion is not available for one of the reasons stated above, you can create a routine. On the transformation rule definition, choose Routine with Unit. The routine editor gives you the additional return parameter UNIT, whose value is used to determine the target unit of measure.

More information: Routines in Transformations [page 511].

4.15.6 ABAP Routines in Transformations

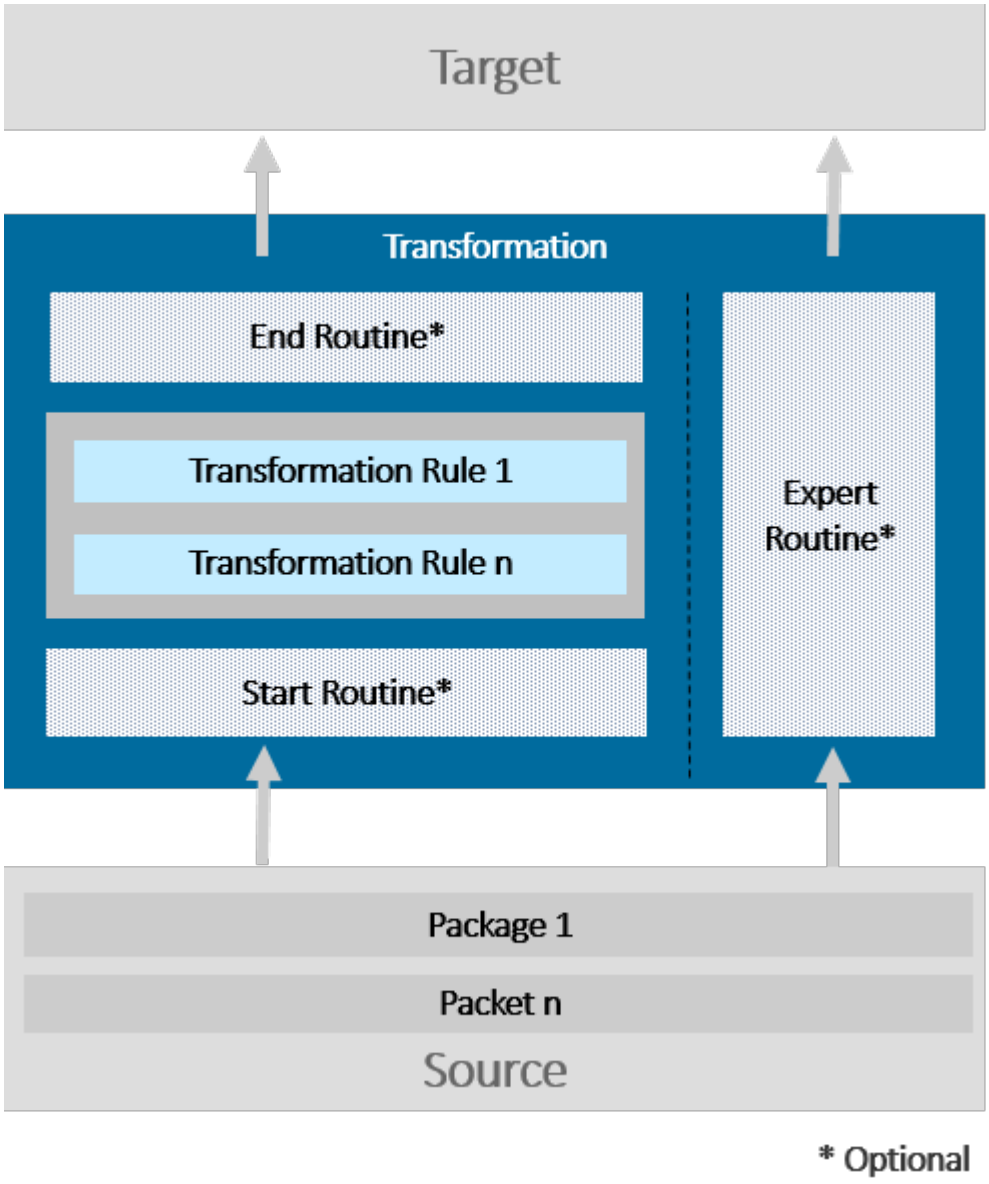
You use routines to define complex transformation rules.

> **Note**
>
> If you want to create ABAP routines, you must select the ABAP runtime under *Runtime Properties*.

ABAP routines are methods of a local ABAP class that consist of a predefined definition area and an implementation area. The TYPES for the inbound and outbound parameters and the signature of the routine (ABAP method) are stored in the definition area. The routine itself is created in the implementation area. ABAP object statements are available in the coding of the routine. Upon generation, the coding is embedded in the local class of the transformation program as the method.

> **Note**
>
> You should not change this class, and you should only call it using the transformation.

The following graphic shows the position of these routines in the data flow:



Features

The routine has a global part and a local part. In the global part, you define global data declarations 'CLASS DATA'. These are available in all routines.

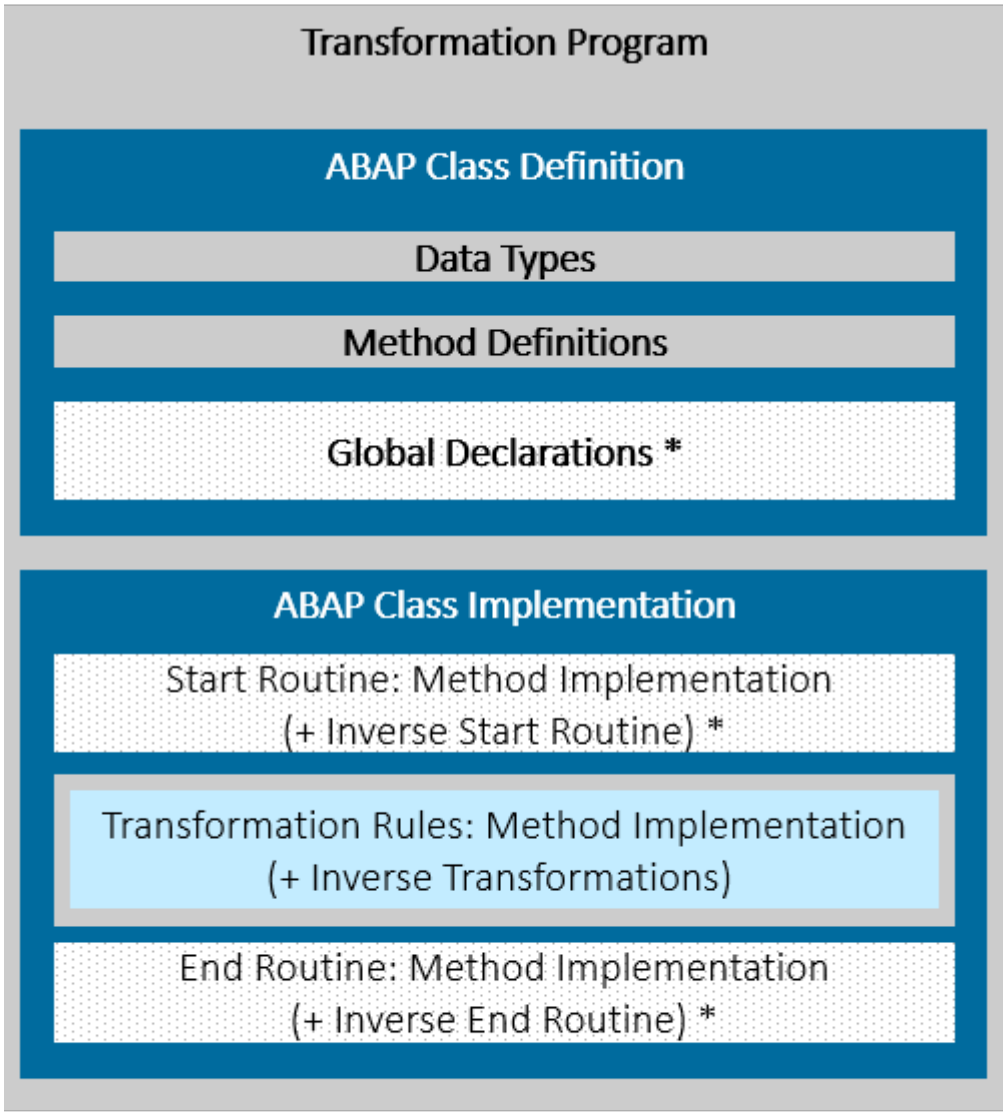
> **Note**
>
> Do not create any global implementations. These cannot be processed. Changes or enhancements outside of the methods cannot be applied and are therefore lost.

> **Note**
>
> In earlier releases you could create declarations in a second global area. The second global area is no longer supported. You have to copy these declarations from the second global area into the first global area. See SAP Note 2654109 for more details.

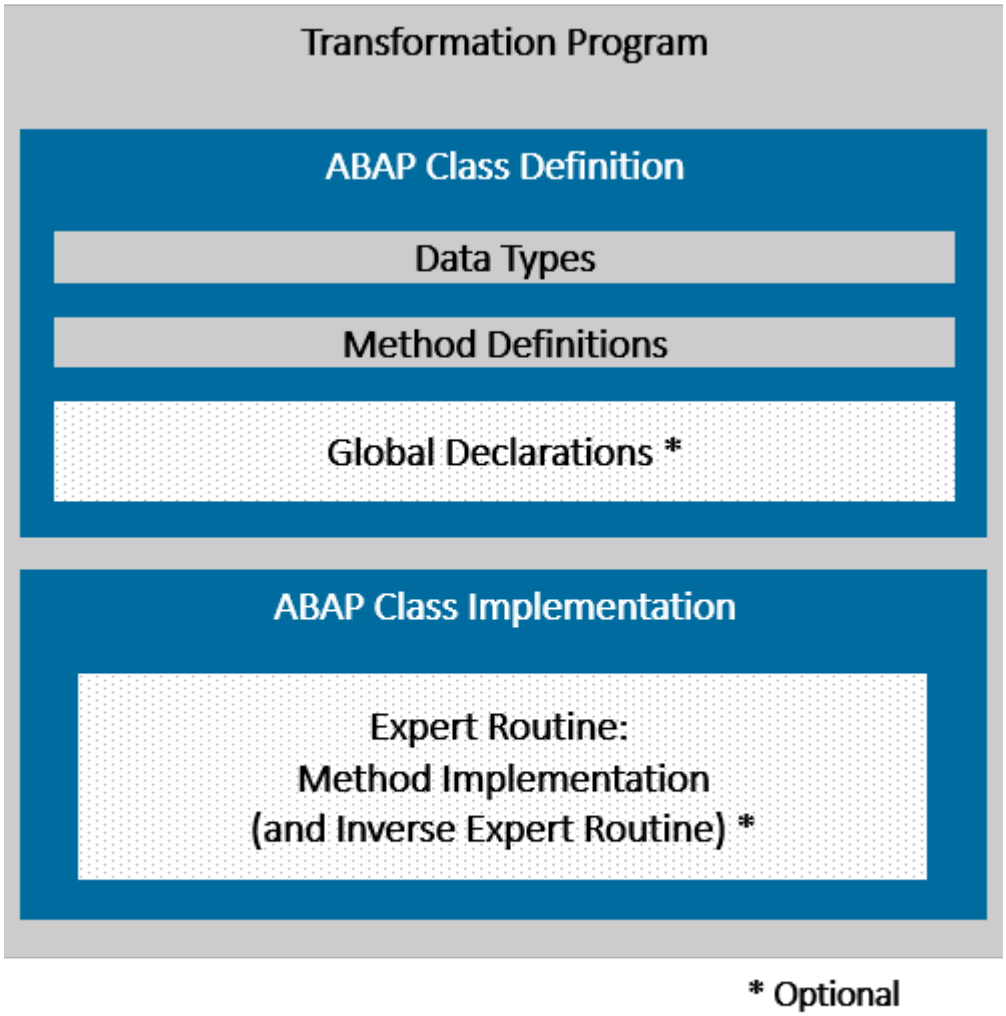
You can create function modules, methods or external subprograms in the ABAP Workbench if you want to reuse source code in routines. You can call these in the local part of the routine. The source code of the method is copied into the metadata of the transformation and transported with the transformation. In the target system, a new class is created from the metadata.

There are various types of routine in the transformation: start routine, routine for key figures or characteristics, end routine and expert routine.

The following figure shows the structure of the transformation program with transformation rules, start routine, and end routine:



The following figure shows the structure of the transformation program with expert routine:



Start Routine

The start routine is run for each data package at the start of the transformation. The start routine has a table in the format of the source structure as input and output parameters. It is used to perform preliminary calculations and store these in a global data structure or in a table. You can access this structure or table from other routines. You can modify or delete data in the data package.

Routine for Key Figures or Characteristics

This routine is available as a rule type; you can define the routine as a transformation rule for a key figure or a characteristic. The input and output values depend on the selected field in the transformation rule. For more information, see the *Routine* section under Rule Type [page 504].

End Routine

An end routine is a routine with a table in the target structure format as an inbound parameter and an outbound parameter. You can use an end routine to post-process data, package-by-package, after transformation. For example, you can delete records that are not to be updated, or perform data checks.

The ABAP end routine can be created independently of the runtime in ABAP or in SAP HANA script, but only in the last transformation that is written to the target.

> ⚠ Caution
>
> If the target of the transformation is a DataStore object, key figures are updated by default with the aggregation behavior *Overwrite* (MOVE). You have to use a dummy rule to override this.

Expert Routine

This type of routine is only intended for use in special cases. You can use the expert routine if the other transformation functions are not sufficient. You can use the expert routine as an interim solution until the necessary functions are available in the standard routine.

If you have already created transformation rules, the system deletes them once you have created an expert routine.

Navigation attributes of the source of the transformation are not available in the expert routine.

> ⚠ Caution
>
> If the target of the transformation is a DataStore object, key figures are updated by default with the aggregation behavior *Overwrite* (MOVE).

More information:

Example of Start Routine [page 518]

Example of Characteristic Routine [page 518]

Example of End Routine [page 520]

4.15.6.1 Create ABAP Routine

Prerequisites

You have created an ABAP in Eclipse project.

Procedure

1. You are in the BW modeling tools, editing your transformation on the *General* tab.
   - To create a start routine, choose *Start Routine* > *Create*.
   - To create an expert routine, choose *Expert Routine* > *Create*.

- To create an end routine, choose *End Routine* > *Create ABAP*.
- To create a characteristic routine, select the characteristic. In the *Properties view*, you can select the rule type *Routine* in the rule details.

2. If you are creating a characteristic routine, the routine editor appears immediately. If you are creating a start routine or an end routine, you can first select fields that are to be updated. Some fields are required for further processing however, meaning that they must be updated. These fields are marked as *mandatory*.
3. You are in the routine editor. To create a routine, enter the following:
   Between `*$*$ begin of global ...` and `*$*$ end of global ...`, you can define 'CLASS DATA' global data declarations. These are available in all routines. Data declarations with 'DATA' can only be accessed in the current package.
   This means that you can use intermediate results in other routines for example, or reuse results when you call a routine again at a later time.

   > **Note**
   >
   > When you perform serial loads, one process instance is used for the entire request. In this case, data with the 'CLASS DATA' data declaration can be accessed for the entire request (all packages).
   >
   > Several process instances are used when you perform parallel loads.
   >
   > A single process instance can be used multiple times. The exact number of times depends on the number of data packages to be processed and the number of process instances that are available. This means that with parallel loads too, data with the 'CLASS DATA' data declaration is not initialized for each data package and may still contain data from predecessor packages.
   >
   > Therefore use 'CLASS DATA' or 'DATA' for the global data, depending on the scenario.

4. Insert your program code for the routine between `*$*$ begin of routine ...` and `*$*$ end of routine ...`.

   > **Note**
   >
   > Editing outside of the two comments is not prevented technically. However, if you enter code outside of the two comments, this coding is lost when the routine is transferred.

   For information about the parameters of the routine, see:
   - Start Routine Parameters [page 520]
   - Routine Parameters for Key Figures or Characteristics [page 521]
   - End Routine Parameters [page 522]

   > **Caution**
   >
   > Do not use an SAP COMMIT in your program code (ABAP statement: COMMIT WORK). When this statement is executed, the cursor used to read from the source is lost. Use a DB COMMIT (call function module DB_COMMIT) instead, or avoid using such COMMITs altogether.

5. Check the syntax of your routine.
6. Save the routine. You end the editing of the routine by leaving the editor.

More information:

Example of Start Routine [page 518]

Example of Characteristic Routine [page 518]

Example of End Routine [page 520]

4.15.6.1.1 Example of a Start Routine

Use

You are loading data using DataSource *General Ledger: Transaction Figures* (FI_GL_1) in the SAP ERP system into DataStore object *FIGL: Transaction Figures* (0FIGL_O06).

You want to create a start routine that deletes all the records from a data package with debit and credit postings that are zero.

1. Create a transformation. The source of the transformation has the *Total Debit Postings* (UMSOL) and *Total Credit Postings* (UMHAB) fields. They are assigned to the InfoObjects *Total Debit Postings* (0DEBIT) and *Total Credit Postings* (0CREDIT).
2. You are in the BW modeling tools, editing your transformation on the *General* tab. Choose *Start Routine* *Create*. The routine editor opens.
3. Go to the local part of the routine. Enter the following code:

   Sample Code

   ```
   *---------------------------------------------------------------------*
     METHOD start_routine.
   *=== Segments ===

       FIELD-SYMBOLS:
         <SOURCE_FIELDS>    TYPE _ty_s_SC_1.

   *$*$ begin of routine - insert your code only below this line        *-*

     DELETE SOURCE_PACKAGE where UMHAB = 0 and UMSOL = 0.

   *$*$ end of routine - insert your code only before this line         *-*
     ENDMETHOD.                    "start_routine
   *---------------------------------------------------------------------*
   ```

   The DELETE statement is the only line you need to filter debit and credit postings without values from the data package.
4. Save the class and go back to the transformation editor.
5. Activate the transformation. The transformation has object status *Active*, but the version status is *Revised*.

4.15.6.1.2 Example of Characteristic Routine

Use

You are loading data using DataSource *General Ledger: Transaction Figures* (0FI_GL_1) in the SAP ERP system into DataStore object *FIGL: Transaction Figures* (0FIGL_O06).

You want to create a routine for the characteristic *Debit/Credit Indicator* (0FI_DBCRIND) in the target that assigns the value **D** to debit postings and the value **C** to credit postings.

1. You are in the BW modeling tools, editing your transformation on the *Rules* tab. Select the InfoObject *Debit / Credit Indicator* (0FI_DBCRIND) and the rule detailsare displayed in the *Properties View*.
2. You choose *Add Fields* and add the *Total Debit Postings* (UMSOL) and *Total Credit Postings* (UMHAB) fields so that they are available in the routine.
3. You choose *Routine* as the rule type. When you choose *Routine* *Edit* the class editor opens.
4. You enter the following lines of code. They return either a **D** or a **C** as the result value:

Sample Code

```
*---------------------------------------------------------------------*

  METHOD compute_0FI_DBCRIND.

    DATA:
      MONITOR_REC    TYPE rsmonitor.

*$*$ begin of routine - insert your code only below this line        *-*

*    result value of the routine
    if SOURCE_FIELDS-umhab ne 0 and SOURCE_FIELDS-umsol eq 0.
      RESULT = 'D'.
    elseif SOURCE_FIELDS-umhab eq 0 and SOURCE_FIELDS-umsol ne 0.
      RESULT = 'C'.
    else.
      monitor_rec-msgid = 'ZMESSAGE'.
      monitor_rec-msgty = 'E'.
      monitor_rec-msgno = '001'.
      monitor_rec-msgv1 = 'ERROR, D/C Indicator'.
      monitor_rec-msgv2 = SOURCE_FIELDS-umhab.
      monitor_rec-msgv3 = SOURCE_FIELDS-umsol.
      append monitor_rec to monitor.
      RAISE EXCEPTION TYPE CX_RSROUT_ABORT.
    endif.

*$*$ end of routine - insert your code only before this line         *-*
  ENDMETHOD.                    "compute_0FI_DBCRIND
*---------------------------------------------------------------------*
```

The system checks whether the debit and credit postings contain values:

- If the debit posting has values that are not equal to zero and the credit posting is equal to zero, the system assigns the value **D**.
- If the credit posting has values that are not equal to zero and the debit posting is equal to zero, the system assigns the value **C**.
- If both the debit and credit postings contain values, the system outputs an error in the monitor and terminates the load process.

5. Save and activate the class, and close the class editor.
6. You save the transformation.

4.15.6.1.3 Example of an End Routine

Use

You are loading data using DataSource *General Ledger: Transaction Figures* (0FI_GL_1) in the SAP ERP system into DataStore object *FIGL: Transaction Figures* (0FIGL_O06). *Transaction Figures* (0FIGL_O06).

You want to create an end routine to fill the additional InfoObject *Plan/Actual Indicator* (ZPLACTUAL). You also want the routine to read the *Value Type* field. If the value is 10 (actual), value A is written to the *Plan/Actual Indicator* InfoObject; if the value is 20 (plan), value P is written to the *Plan/Actual Indicator* InfoObject.

1. You are in the BW modeling tools, editing your transformation on the *General* tab. Choose *End Routine* > *Create ABAP*. The routine editor opens.
2. You enter the following lines of code:

   Sample Code

   ```
   *---------------------------------------------------------------------*
     METHOD end_routine.
   *=== Segments ===

       FIELD-SYMBOLS:
         <RESULT_FIELDS>    TYPE _ty_s_TG_1.

   *$*$ begin of routine - insert your code only below this line        *-*

       loop at RESULT_PACKAGE assigning <RESULT_FIELDS>
         where vtype eq '010' or vtype eq '020'.
         case <RESULT_FIELDS>-vtype.
           when '010'.
             <RESULT_FIELDS>-/bic/zplactual = 'A'. "Actual
           when '020'.
             <RESULT_FIELDS>-/bic/zplactual = 'P'. "Plan
         endcase.
       endloop.
   *$*$ end of routine - insert your code only before this line         *-*
     ENDMETHOD.                    "end_routine
   *---------------------------------------------------------------------*
   ```

   The code loops through the result_package searching for values that have the value type 10 or 20. For these values, the appropriate value is passed on to InfoObject *Plan/Actual Indicator* (ZPLACTUAL).
3. Save the class and go back to the transformation editor.
4. Activate the transformation. The transformation has object status *Active*, but the version status is *Revised*.

4.15.6.2 Start Routine Parameters

Importing

- REQUEST: Request ID
- DATAPAKID: Number of the current data package
- SEGID: Number of the segment

Exporting

- MONITOR: Table for user-defined monitoring. This table is filled using row structure MONITOR_REC (the record number of the processed record is inserted automatically from the framework).

Changing:

- SOURCE_PACKAGE: Structure that contains the inbound fields of the routine.

Raising:

- CX_RSROUT_ABORT: If a raise exception type cx rsrout_abort is triggered in the routine, the system terminates the entire load process. The request is highlighted in the extraction monitor as having been terminated. The system stops processing the current data package. This can be useful if serious errors occur.
- CX_RSBK_ERRORCOUNT: If a raise exception type cx rsbk_errorcount is triggered in the routine, the entire processing of the data package is terminated.

4.15.6.3 Routine Parameters for Key Figures or Characteristics

Importing

- REQUEST: Request ID
- DATAPAKID: Number of the current data package
- SOURCE_FIELDS: Structure with the routine source fields defined on the UI
- SEGID: Number of the segment

Exporting

- MONITOR: Table for user-defined monitoring. This table is filled using row structure MONITOR_REC (the record number of the processed record is inserted automatically from the framework).
- RESULT: You have to assign the result of the computed key figure or computed characteristic to the RESULT variables.
- CURRENCY (optional): If the routine has a currency, you have to assign the currency here.
- UNIT (optional): If the routine has a unit, you have to assign the unit here.

Raising

Exception handling by means of exception classes is used to control what is written to the target:

- CX_RSROUT_SKIP_RECORD: If a `raise exception type cx_rsrout_skip_record` is triggered in the routine, the system stops processing the current row and continues with the next data record.
- CX_RSROUT_SKIP_VAL: If an `exception type cx_rsrout_skip_val` is triggered in the routine, the target field is deleted.
- CX_RSROUT_ABORT: If a `raise exception type cx rsrout_abort` is triggered in the routine, the system terminates the entire load process. The request is highlighted in the extraction monitor as having been *terminated*. The system stops processing the current data package. This can be useful if serious errors occur.

4.15.6.4 End Routine Parameters

Importing

- REQUEST: Request ID
- DATAPAKID: Number of the current data package
- SEGID: Number of the segment

Exporting

- MONITOR: Table for user-defined monitoring. This table is filled using row structure MONITOR_REC (the record number of the processed record is inserted automatically from the framework).

Changing

- MONITOR: Table for user-defined monitoring. This table is filled using row structure MONITOR_REC (the record number of the processed record is inserted automatically from the framework).

Raising

- CX_RSROUT_ABORT: If a raise exception type cx rsrout_abort is triggered in the routine, the system terminates the entire load process. The request is flagged as terminated in the extraction monitor. The system stops processing the current data package. This can be useful if serious errors occur.
- CX_RSBK_ERRORCOUNT: If a raise exception type cx rsbk_errorcount is triggered in the routine, the entire processing of the data package is terminated.

> ⚠ Caution
>
> If only the key fields are updated for master data attributes, all the attributes are initialized, regardless of the settings described here. For more information, see 1096307.

> ⓘ Note
>
> If key fields are filled in the end routine, these fields are also filled in the DataStore object. This occurs regardless of the settings for the field list of the end routine.

4.15.6.5 Creating Inversion Routines

Using inversion routines make sense for virtual providers and to improve performance by reducing the size of requests to large amounts of data. Inversion routines convert the selection criteria of a query into selection criteria for the extractor. Without these selection criteria, all values of the source are selected.

You also have to create an inversion routine when calling up a transaction in another SAP system using the report-report interface if a routine is used in the transformation, as the selections cannot otherwise be transferred to the source system.

You can create an inversion routine for all types of routine. The following rules apply:

- With expert routines, there is no segmentation into conditions.
- With start routines, the system performs segmentation into conditions. The system applies this to the complete source structure. The source structure is the start and end point.

- With end routines, the target structure is the start and end point.

> **Note**
>
> If SAP HANAis set as the runtime, no inverse routine is available for ABAP end routines.SAP HANA

Prerequisites

You have already created a routine. On the *General* tab in the editing screen for your transformation, you have set the *Create Inverse Routines for ABAP Routines*flag.

Procedure

You are in the routine editor. To create an inversion routine, enter the following:

1. Insert your program code for the inversion routine between **$ *$ begin of routine - insert your code only below this line *-** and **$ *$ end of routine - insert your code only before this line *-**.
   With an inversion routine for a jump using the report-report interface, you have to make an exact inversion so that the selections can be transferred exactly.
   For information about the parameters of the routine, see Inversion Routine Parameters [page 523]
2. Check the syntax of your routine.
3. Save the class and close the class editor.

Example

Here is an example of an inversion routine: Example of Inversion Routine [page 524]

4.15.6.5.1 Inversion Routine Parameters

The inversion routine has method *invert*.

It has the following parameters:

Importing

- i_th_fields_outbound: Fields/InfoObjects for the query structure
- i_r_selset_outbound: Query selection conditions
- i_is_main_selection: Allows you to transfer complex selection conditions such as selection conditions for columns.
- i_r_selset_outbound_complete: All selections
- i_r_universe_inbound: Description of source structure with regard to set objects.

Changing

- c_th_fields_inbound: Fields/InfoObjects for the target structure
- c_r_selset_inbound: Taget selection conditions. You can fill the target field from more than one source field. In this case, you have to define more than one condition.
- c_exact: Allows you to specify whether you want the transformation of the selection criteria to be performed exactly. If the condition can be filled exactly, a direct call is possible. This is important when you call the report-report interface. If the condition cannot be filled exactly, a selection screen appears for the user.

4.15.6.5.2 Example of Inversion Routine

Use

In this example, the German keys 'HERR' and 'FRAU' in the target characteristic are mapped to the English keys 'MR' and 'MRS' from the PASSFORM field (form of address) in the source. All other values from the source field are mapped to the initial value.

The coding of the routine is as follows:

Sample Code

```
*---------------------------------------------------------------------*
*       CLASS routine DEFINITION
*---------------------------------------------------------------------*
*
*---------------------------------------------------------------------*
CLASS lcl_transform DEFINITION.
  PUBLIC SECTION.

    TYPES:
      BEGIN OF _ty_s_SC_1,
*      Field: PASSFORM Anrede.
        PASSFORM           TYPE C LENGTH 15,
      END   OF _ty_s_SC_1.
    TYPES:
      BEGIN OF _ty_s_TG_1,
*      InfoObject: 0PASSFORM Anrede.
        PASSFORM           TYPE /BI0/OIPASSFORM,
      END   OF _ty_s_TG_1.
  PRIVATE SECTION.

    TYPE-POOLS: rsd, rstr.

*$*$ begin of global - insert your declaration only below this line  *-*
    DATA p_r_set_mr    TYPE REF TO cl_rsmds_set.
    DATA p_r_set_mrs   TYPE REF TO cl_rsmds_set.
    DATA p_r_set_space TYPE REF TO cl_rsmds_set.

*$*$ end of global - insert your declaration only before this line   *-*
    METHODS
      compute_0PASSFORM
        IMPORTING
          request                type rsrequest
          datapackid             type rsdatapid
```

```
          SOURCE_FIELDS              type _ty_s_SC_1
        EXPORTING
          RESULT                   type _ty_s_TG_1-PASSFORM
          monitor                  type rstr_ty_t_monitor
        RAISING
          cx_rsrout_abort
          cx_rsrout_skip_record
          cx_rsrout_skip_val.
    METHODS
      invert_0PASSFORM
        IMPORTING
          i_th_fields_outbound         TYPE rstran_t_field_inv
          i_r_selset_outbound          TYPE REF TO cl_rsmds_set
          i_is_main_selection          TYPE rs_bool
          i_r_selset_outbound_complete TYPE REF TO cl_rsmds_set
          i_r_universe_inbound         TYPE REF TO cl_rsmds_universe
        CHANGING
          c_th_fields_inbound          TYPE rstran_t_field_inv
          c_r_selset_inbound           TYPE REF TO cl_rsmds_set
          c_exact                      TYPE rs_bool.
ENDCLASS.                    "routine DEFINITION
*$*$ begin of 2nd part global - insert your code only below this line  *
... "insert your code here
*$*$ end of 2nd part global - insert your code only before this line   *

*---------------------------------------------------------------------*
*       CLASS routine IMPLEMENTATION
*---------------------------------------------------------------------*
*
*---------------------------------------------------------------------*
CLASS lcl_transform IMPLEMENTATION.

  METHOD compute_0PASSFORM.

*   IMPORTING
*     request     type rsrequest
*     datapackid  type rsdatapid
*     SOURCE_FIELDS-PASSFORM TYPE C LENGTH 000015
*    EXPORTING
*      RESULT type _ty_s_TG_1-PASSFORM

    DATA:
      MONITOR_REC    TYPE rsmonitor.

*$*$ begin of routine - insert your code only below this line        *-*

    CASE SOURCE_FIELDS-passform.
      WHEN 'HERR'. RESULT = 'MR'.
      WHEN 'FRAU'. RESULT = 'MRS'.
      WHEN OTHERS. RESULT = space.
    ENDCASE.

*$*$ end of routine - insert your code only before this line         *-*
  ENDMETHOD.                    "compute_0PASSFORM
```

The corresponding inversion routine is as follows:

Sample Code

```
*$*$ begin of inverse routine - insert your code only below this line*-*
    DATA l_r_set        TYPE REF TO cl_rsmds_set.
    IF i_r_selset_outbound->is_universal( ) EQ rsmds_c_boolean-true.

*     If query requests all values for characteristic 0PASSNAME
*     request also all values from source field PASSNAME
```

```
      c_r_selset_inbound = cl_rsmds_set=>get_universal_set( ).
      c_exact = rs_c_true. "Inversion is exact

    ELSE.
      TRY.
          IF me->p_r_set_mrs IS INITIAL.
*           Create set for condition PASSFORM = 'FRAU'
            me->p_r_set_mrs = i_r_universe_inbound-
>create_set_from_string( 'PASSFORM = ''FRAU''' ).
          ENDIF.
          IF me->p_r_set_mr IS INITIAL.
*           Create set for condition PASSFORM = 'HERR'
            me->p_r_set_mr = i_r_universe_inbound-
>create_set_from_string( 'PASSFORM = ''HERR''' ).
          ENDIF.
          IF me->p_r_set_space IS INITIAL.
*           Create set for condition NOT ( PASSFORM = 'FRAU' OR PASSFORM =
'HERR' )
            l_r_set           = me->p_r_set_mr->unite( me->p_r_set_mrs ).
            me->p_r_set_space = l_r_set->complement( ).
          ENDIF.

*         Compose inbound selection
          c_r_selset_inbound = cl_rsmds_set=>get_empty_set( ).
*         Check if outbound selection contains value 'MR'
          IF i_r_selset_outbound->contains( 'MR'  ) EQ rsmds_c_boolean-true.
            c_r_selset_inbound = c_r_selset_inbound->unite( me->p_r_set_mr ).
          ENDIF.
*         Check if outbound selection contains value 'MRS'
          IF i_r_selset_outbound->contains( 'MRS' ) EQ rsmds_c_boolean-true.
            c_r_selset_inbound = c_r_selset_inbound->unite( me->p_r_set_mrs ).
          ENDIF.
*         Check if outbound selection contains initial value
          IF i_r_selset_outbound->contains( space ) EQ rsmds_c_boolean-true.
            c_r_selset_inbound = c_r_selset_inbound->unite( me-
>p_r_set_space ).
          ENDIF.
          c_exact = rs_c_true. "Inversion is exact

        CATCH cx_rsmds_dimension_unknown
              cx_rsmds_input_invalid
              cx_rsmds_sets_not_compatible
              cx_rsmds_syntax_error.

*         Normally, should not occur
*         If the exception occurs request all values from source
*         for this routine to be on the save side
          c_r_selset_inbound = cl_rsmds_set=>get_universal_set( ).
          c_exact = rs_c_false. "Inversion is no longer exact

      ENDTRY.
    ENDIF.

*   Finally, add (optionally) further code to transform outbound projection
*   to inbound projection
*   Check if outbound characteristic 0PASSFORM (field name PASSFORM)
*   is requested for the drilldown state of the query
    READ TABLE i_th_fields_outbound
         WITH TABLE KEY segid     = 1           "Primary segment
                        fieldname = 'PASSFORM'
         TRANSPORTING NO FIELDS.
    IF sy-subrc EQ 0.
*     Characteristic 0PASSFORM is needed
*     ==> request (only) field PASSFORM from the source for this routine
      DELETE c_th_fields_inbound
        WHERE NOT ( segid     EQ 1          OR
                    fieldname EQ 'PASSFORM'    ).
    ELSE.
```

```
*     Characteristic 0PASSFORM is not needed
*     ==> don't request any field from source for this routine
      CLEAR c_th_fields_inbound.
    ENDIF.

*$*$ end of inverse routine - insert your code only before this line *-*
  ENDMETHOD.                    "invert_0PASSFORM
ENDCLASS.                    "routine IMPLEMENTATION
```

4.15.6.5.3 Details for Implementing the Inversion Routine

Use

Set Objects

The purpose of an inverse transformation is to convert selection conditions of the query that are formulated for the target of the transformation (outbound) into selection conditions for the source (inbound). To do this, the selection conditions are converted into a multidimensional set object. In ABAP objects, these are the instances of class CL_RSMDS_ST). The advantage of this representation is that set operations (intersection, union, and complement), that can only be processed with a great deal of effort with the usual RANGE table representation, can now be processed easily.

Universes

There are always two uniquely defined trivial instances of class CL_RSMDS_SET that represent the empty set and the total set (that is, all the values). You can recognize these instances from the result RSMDS_C_BOOLEAN-TRUE of the functional methods IS_EMPTY and IS_UNIVERSAL. All other instances are always assigned to a *Universe* (instance of class CL_RSMDS_UNIVERSE) and return the result RSMDS_C_BOOLEAN-TRUE for the specified methods. You can obtain the reference of the assigned universe for non-trivial instances of class CS_RSMDS_SET with method GET_UNIVERSE. This method returns an initial reference for these two trivial instances since the universe is not uniquely defined in this case.

A universe represents the sum of all the dimensions (represented by instances of the interface IF_RSMDS_DIMENSION). A dimension is always uniquely defined by a dimension name in the universe. With method GET_DIMENSION_BY_NAME in class CL_RSMDS_UNIVERSE, you can obtain a dimension reference using the unique dimension name. The dimension name is generally the same as the field name in a structure. There are different types of universe in the system (subclasses of class CL_RSMDS_UNIVERSE). The dimensions have different meanings. For example, a dimension corresponds to an InfoObject in class CL_RS_INFOOBJECT_UNIVERSE. In the case of InfoObjects, you have the two methods IOBJNM_TO_DIMNAME and DIMNAME_TO_IOBJNM that transform an InfoObject name into a dimension name or a dimension name into an InfoObject name. For an InfoObject-based universe, there is exactly one instance (singleton) that contains (nearly) all the active InfoObjects in the system as dimensions (with the exception of InfoObjects in InfoSets). This instance is returned with the method GET_INSTANCE of class CL_RS_INFOOBJECT_UNIVERSE.

In the case of DataSources, there is a uniquely defined universe for each combination of logical system name (I_LOGSYS), DataSource name (I_DATASOURCE) and segment ID (I_SEGID). You can find the reference of the universe with the method CREATE_FROM_DATASOURCE_KEY of class CL_RSDS_DATASOURCE_UNIVERSE. The initial segment ID always provides the primary segment, which normally is the only segment on which selection conditions can be formulated for a source and accepted. All the fields in the DataSource segment that are selected for direct access form the dimensions of a DataSource universe with the same name.

Here, too, you get a dimension reference (instance for interface IF_RSMDS_DIMENSION) with the method GET_DIMENSION_BY_NAME of the universe.

If you want to project a selection to a given dimension from a general selection, that is for any instance of the class CL_RSMDS_SET, you first need a reference to the universe to which the instance belongs (method GET_UNIVERSE, see above). You get the dimension reference from the reference to the universe using the dimension/field name from method GET_DIMENSION_BY_NAME. With the dimension reference, you can then project a representation for a one-dimensional condition using method TO_DIMENSION_SET. You can then convert a one-dimensional projection into an Open SQL or RANGE condition for the corresponding field with the methods TO_STRING and TO_RANGES. Vice versa, you can create an instance on the dimension reference for a one-dimensional set object from a RANGE table using the method CREATE_SET_FROM_RANGES. The SIGNs 'I' and 'E' as well as the OPTIONs 'EQ', 'NE', 'BT', 'NB', 'LE', 'GT', 'LT', 'GE', 'CP' and 'NP' are supported. There are only restrictions for 'CP' and 'NP'. These may only be used for character-type dimensions/fields and may only contain the masking character'*', which must always be at the end of the character chain. For example, 'E' 'NP' 'ABC*' is a valid condition, but 'I' 'CP' '*A+C*' is not.

Using method GET_DIMENSIONS in class CL_RSMDS_SET, you can get a table with the references of all dimensions that are restricted in the corresponding instance of the set object. With the method GET_NAME, you can get the unique dimension name for each dimension reference in the table that is returned. In this way you can check if there is a restriction for a given InfoObject or field. It can be projected as described above.

With the universe reference, you can create an instance for a set object (especially for multidimensional set objects) from an Open SQL expression. In the Open SQL expression that is passed, the "field names" must be the valid dimension names in the universe. You may use elementary conditions with the comparison operators '=', '<>', '<=', '>', '<' und '>=' in the Open SQL expression. The left side must contain a valid dimension name and the right side must contain a literal that is compatible with the data type of the dimension. You can also use elementary conditions with 'BETWEEN', 'IN' and 'LIKE' with the appropriate syntax. Elementary conditions may be linked with the logical operators 'NOT', 'AND' and 'OR' to create complex conditions. You may also use parentheses to change the normal order of evaluation ('NOT' is stronger than 'AND', 'AND' is stronger than 'OR').

With the method CREATE_SET_FROM_RANGES of the universe reference, you can also directly create a set object for a multidimensional condition. To do this, the internal table passed in I_T_RANGES must contain a RANGE structure (with the components SIGN, OPTION, LOW and HIGH) in its row structure and must also have an additional component for a dimension name. Parameter I_FIELDNAME_DIMENSION must pass the name of these components to method CREATE_SET_FROM_RANGES.

You can always create an instance for the complementary condition for any instance of the class CL_RSMDS_SET using the functional method.

If two instances of the class CL_RSMDS_SET belong to the same universe, you can create an instance for the intersection or union by passing the other instance as parameter I_R_SET when you call the functional method INTERSECT or UNITE.

With the method TRANSFORM, you can also transform an instance of a set object into an instance of a set object of another universe. If required, you can thus perform a projection or assign dimension names in a different manner. These methods are recommended, for example if the name of the source field differs from the name of the target field within the transformation. You can pass a reference to the target universe to the method in the optional parameter I_R_UNIVERSE. If the parameter remains initial, the system assumes that the source and target universes are identical. With parameter I_TH_DIMMAPPINGS you can map the dimension names of the source universe (component DIMNAME_FROM) in different dimension names on the target universe (component DIMNAME_TO). If component DIMNAME_TO remains initial, a restriction of the

source dimension (in DIMNAME_FROM) is not transformed into a restriction of the target universe. As a result, there is a projection. The following mapping table

DIMNAME_FROM	DIMNAME_TO
AIRLINEID	CARRID
CONNECTID	CONNID
FLIGHTDATE	

transforms a set object that corresponds to the Open SQL condition

AIRLINEID = 'LH' AND CONNECTID = '0400' AND FLIGHTDATE = '20070316' OR

AIRLINEID = 'DL' AND CONNECTID = '0100' AND FLIGHTDATE = '20070317'

into a set object that corresponds to the Open SQL condition

CARRID = 'LH' AND CONNID = '0400' OR

CARRID = 'DL' AND CONNID = '0100',

for example.

Start and End Routines

Parameters I_R_SELSET_OUTBOUND and I_R_SELSET_OUTBOUND_COMPLETE are passed to the start and end routines for the transformation of the selection conditions. The references passed in the two parameters are identical for simple queries, and parameter I_IS_MAIN_SELECTION is defined by the constant RS_C_TRUE. For complex queries that, for example contain restricted key figures or structure elements with selections, the inverse start routine is called several times. The first time, I_R_SELSET_OUTBOUND is called with the restrictions from the global filter and the restrictions that are shared by all structure elements. In this call, parameter I_IS_MAIN_SELECTION is also set to RS_C_TRUE. There are further calls with selections for the specific structure element. However, they are combined so that they no longer overlap. In these calls, I_IS_MAIN_SELECTIN is set to RS_C_FALSE. The complete selection condition is contained in I_R_SELSET_OUTBOUND_COMPLETE for all calls. In order to transform the selections exactly in the start and end routines, the transformation of I_R_SELSET_OUTBOUND into a set object C_R_SELSET_INBOUND in the universe of the source structure (is passed as a reference with parameter I_R_UNIVERSE_INBOUND) must be made exactly for each call. This must be documented by returning the value RS_C_TRUE in parameter C_EXACT.

Expert Routines

Parameter I_R_SELSET_OUTBOUND always passes the complete selections of the target to the expert routine. The expert routine must return a complete selection for the source in C_R_SELSET_INBOUND. As previously for the start and end routines, it could be advantageous to break down a complex selection S into a global selection G and several disjoint subsections Ti (i = 1...n). You can break down the passed reference with the method GET_CARTESIAN_DECOMPOSITION. You can see the global selection in parameter E_R_SET; the subselections are entries in the internal table that is returned in parameter E_TR_SETS. For the decomposition, the following is always applicable:

$$S = G \cap (T_1 \cup ... \cup T_n)$$

and

$$T_i \cap T_i = \varnothing \text{ für } i \neq j.$$

You should invert the global selection and each subselection individually

$$(G \rightarrow G', T_i \rightarrow T_i')$$

and compose the inverted results again in the form

$$G' \cap (T_1' \cup ... \cup T_n')$$

Usually you can only ensure an exact inversion of a complex selection condition by using this type of decomposition. If the method GET_CARTESIAN_DECOMPOSITION is called with I_REDUCED = RSMDS_C_BOOLEAN-FALSE, the following is already valid for the decomposition

$$S = (T_1 \cup ... \cup T_n).$$

It is no longer applicable if the call is made with I_REDUCED = RSMDS_C_BOOLEAN-TRUE and

$$(T_1 \cup ... \cup T_n)$$

is usually a superset of S. In this case, the selections Ti are usually simpler.

Passing the Selection Conditions

If the transformed selection conditions for the source return exactly the data records that satisfy the selection conditions of the target after execution of the transformation, then the inverse transformation is considered to be exact. This will not always be possible. For this reason a transformation that is not exact may provide more data records than are needed to satisfy the selection conditions of the target. You can ensure that the results are exact by filtering them with the selection conditions of the target. An inverse transformation, however, should not create a selection condition for the source that selects fewer data records from the source than are needed to satisfy the selection condition of the target.

An inverse transformation that is not exact is indicated by the return value RS_C_FALSE in parameter C_EXACT for at least one inverse routine run. This only has an effect on the performance for queries on the analytic manager (OLAP) since they are always filtered again there. However, in the RSDRI interface, in transaction LISTCUBE, and in function *Display Data* in the context menu of a VirtualProvider, there is no further filtering and the superfluous records are returned or displayed. The property of being exact for an inverse transformation otherwise only has an effect if it is called in the report-report interface. An inversion that is not exact always causes the selection screen to be displayed before the target transaction is executed. This gives the user the chance to check the selections again and to correct them if necessary.

An inverse routine that is not implemented always requests all the values for all the source fields of this routine. Accordingly, parameters C_R_SELSET_INBOUND and C_EXACT always contain an instance for the "All Values" condition or the value RS_C_FALSE when they are called.

> ⓘ Note
>
> One final comment. Selections are always stored in a normed manner in a set object. This means, for example, that the two Open SQL expressions
>
> CARRID = 'LH' AND FLDATE < '20070101'
>
> and
>
> CONNID <= '20061231' AND CARRID = 'LH'
>
> have the same representation as the set object. If you call all the methods that cause the creation of a set object as result with the parameter I_FINAL = RSMDS_C_BOOLEAN-TRUE (this should normally be the default value), you must also make sure that the two objects are identical in the above case (that is, they should have the same references). To check if two instances of the class CL_RSMDS_SET represent the same selection condition, however, you should nevertheless use the method IS_EQUAL and check against the result RSMDS_C_BOOLEAN-TRUE.

4.15.6.6 Regular Expressions in Routines

You can use regular expressions in routines.

A regular expression (abbreviation: RegExp or Regex) is a pattern of literal and special characters which describes a set of character strings. In ABAP, you can use regular expressions in statements FIND and REPLACE, and in classes CL_ABAP_REGEX and CL_ABAP_MATCHER. For more information, see the ABAP key word documentation in the ABAP Editor. This documentation describes the syntax of regular expressions. You can test regular expressions here.

This section provides sample code to illustrate how you can use regular expressions in routines.

Sample Code

```
REPORT z_regex.

DATA: l_input TYPE string,
  l_regex TYPE string,
  l_new  TYPE string.

* Example 1: Insert thousand separator

l_input = '12345678'.
l_regex = '([0-9])(?=([0-9]{3})+(?![0-9]))'.
l_new = '$1,'.

WRITE: / 'Before:', l_input.  "12345678

REPLACE
  ALL OCCURRENCES OF
  REGEX l_regex
  IN l_input WITH l_new.

WRITE: / 'After:', l_input.  "12,345,678

* Example 2: Convert date in US format to German format

l_input = '6/30/2005'.
l_regex = '([01]?[0-9])/([0-3]?[0-9])/'.
l_new = '$2.$1.'.

WRITE: / 'Before:', l_input.  "6/30/2005

REPLACE
  ALL OCCURRENCES OF
  REGEX l_regex
  IN l_input WITH l_new.

WRITE: / 'After:', l_input.  "30.6.2005

* Example 3: Convert external date in US format to internal date

DATA: matcher  TYPE REF TO cl_abap_matcher,
  submatch1 TYPE string,
  submatch2 TYPE string,
  match  TYPE c.

l_input = '6/30/2005'.
l_regex = '([01]?)([0-9])/([0-3]?)([0-9])/([0-9]{4})'.

matcher = cl_abap_matcher=>create( pattern  = l_regex
  text  = l_input ).

match = matcher->match( ).

TRY.
  CALL METHOD matcher->get_submatch
  EXPORTING
  index  = 1
  RECEIVING
  submatch = submatch1.
  CATCH cx_sy_matcher.
ENDTRY.

TRY.
  CALL METHOD matcher->get_submatch
  EXPORTING
  index  = 3
  RECEIVING
```

```
  submatch = submatch2.
  CATCH cx_sy_matcher.
ENDTRY.

IF submatch1 IS INITIAL.
  IF submatch2 IS INITIAL.
  l_new = '$5\0$2\0$4'.
  ELSE.
  l_new = '$5\0$2$3$4'.
  ENDIF.
ELSE.
  IF submatch2 IS INITIAL.
  l_new = '$5$1$2\0$4'.
  ELSE.
  l_new = '$5$1$2$3$4'.
  ENDIF.
ENDIF.

WRITE: / 'Before:', l_input.  "6/30/2005

REPLACE
  ALL OCCURRENCES OF
  REGEX l_regex
  IN l_input WITH l_new.

WRITE: / 'After:', l_input.  "20050630
```

4.15.7 SAP HANA SQLScript Routines in the Transformation

In the SAP HANA runtime, you can create SAP HANASQLScript routines.

If you want to create complex transformation rules, for which the existing rule types are not sufficient, you can define SAP HANA SQLScript routines for them.

Note the differences between the two runtimes: Differences Between ABAP Runtime and SAP HANA Runtime [page 500]

You can create SAP HANASQLScript routines as SQLScript. They are then created as a SAP HANAprocedure that is implemented as a method in an AMDP (ABAP managed database procedure) class. The AMDP class is generated by the BW framework and can only be edited in the ABAP Development tools for SAP NetWeaver (ADT).

Related Information

Create SAP HANA SQLScript Routines [page 534]

4.15.7.1 Create SAP HANA SQLScript Routines

In the SAP HANA runtime, you can create SQLScript routines. SAP HANA

Context

You have created an ABAP project for processing the AMDP ABAP class.

Procedure

1. You are in the BW modeling tools, editing your transformation on the *General* tab.
   - To create a start routine, choose *Start Routine* > *Create*.
   - To create an expert routine, choose *Expert Routine* > *Create*.
   - To create an end routine, choose either *End Routine* > *Create SAP HANA Script* or *End Routine* > *Create ABAP*. You can also create an ABAP end routine in a transformation with the SAP HANAruntime. This only applies to transformations that have a persistent target. For other serially activated data flows, an ABAP routine can only be created for the last transformation.
   - To create a characteristic routine, select the characteristic. In the *Properties view*, you can select the rule type *Routine* in the rule details .
2. You can automatically initialize fields in SAP HANA SQLScript routines that contain null values. The null values are then initialized when the routine is executed. If the flag is not set, and there are fields containing null values in the routine, a runtime error occurs, since SAP HANA routines do not allow null values.
3. For start and end routines: Select the fields for the routine.
4. Select an ABAP project. The AMDP class is opened in the ABAP Development Tools for SAP NetWeaver (ADT).
5. Insert your program code.

   > ⓘ Note
   >
   > Make your changes only within the method. All changes outside of the method are not transported and will be overwritten with the next activation.

6. Check the syntax of your routine.

   For creating breakpoints, see SAP Note 2659814

7. Activate the AMDP class. You end the editing of the routine when you leave the editor.

Related Information

SAP HANA SQLScript Reference

4.15.7.2 The ADMP Class

When creating a SAP HANA SQLScript routine, an ABAP class is generated.

The class implements the marker interface IF_AMDP_MARKER_HDB of the ABAP managed database procedure (AMDP). The interface marks the ABAP class as an ADMP class. Methods of an ADMP class can be written as a dabase procedure. The AMDP framework thereby creates a definition of a SAP HANA-specific database procedure for the method. The method name of a field routine is made up of segment ID, group ID and rule ID.

> **Example**
>
> If for example the segment ID is *S001*, the group ID *G01*, and the rule ID *R8*, the method name will be *S0001_G01_R8*.

Related Information

AMDP - SQLScript for the SAP HANA Database

4.16 Creating a Data Transfer Process

You use the data transfer process (DTP) to transfer data from a source to a target object. You can also use the data transfer process to access InfoProvider data in the source directly.

Prerequisites

You have used transformations to define the data flow between the source and target object.

Procedure

1. You are in the BW Modeling tools. You can create a new data transfer process here by choosing *New* > *Data Transfer Process*.
   - from the *File* menu
   - from the context menu of your BW project
   - in the *BW repository* tree, from the context menu on the target object of the data transfer process or on the transformation

   Alternatively, you can create the data transfer process via the data flow.
   You are guided through the creation process by a wizard. More information: Creating a Data Transformation Process Using the Wizard [page 536]

Once the wizard is finished, the data transfer process editor appears.

2. Edit the properties of the data transfer process on the tab pages.
   More Information:
   Editing General Properties [page 537]
   Editing Extraction Properties [page 539]
   Editing Update Properties [page 540]
   Editing Runtime Properties [page 544]
3. Check, save, and activate the data transfer process.
4. If you have activated error handling, go to the *Update* tab page and create an error DTP by pressing .

Results

The data transfer process can now be executed. If you want to control execution using a process chain, you can insert the data transfer process into a chain in the process chain editor. Under *Start execution of data transfer process* in the data transfer editor, you can choose other execution options.

Related Information

Data Transfer Process [page 190]
Excuting Data Transfer Process [page 545]

4.16.1 Creating a Data Transformation Process Using the Wizard

A wizard will now guide you through the processs for creating a data transfer process. In the wizard, select the target and the source of the data transfer process. The system then displays the Editor, where you edit the properties of the data transfer process.

Procedure

In the *Project Explorer* view in the *BW Repository* tree of your BW project, you have chosen *New* > *Data Transfer Process* from the context menu for your target object. You are in the wizard for creating the data transfer process in the first wizard step *New Data Transfer Process*. The BW project and the target object are already selected.

1. Select the source object in *Source*.
   a. Select the object type.
   b. Select the source object.

By choosing *Browse*, you can select the object from the complete list of objects in the system.

By choosing *Browse Paths*, you can display the paths that lead to objects that are already connected as sources with the target object. By selecting a path, you can set the source object.

To select a DataSource, you can also first specify the source system.

2. Choose *Finish*. The editor appears.

4.16.2 Editing General Properties

On the *General* tab, you set the general properties for the data transfer process (DTP). This includes general properties for extraction and properties for executing the data transfer process.

Procedure

General

1. You can change the description.
2. You can call the editor for the source or the target of the data transfer process.

General Extraction Settings - Extraction Settings

3. *Extraction Mode*: Choose Extraction Mode.

   You can choose *Delta* or *Full* mode. To transfer non-cumulative data from ODP source systems with ODP context *SAPI* or from file source systems, you can use extraction mode *Initial Non-Cumulative for Non-Cumulative Values* if the source of the data transfer process is a DataSource that builds an initial non-cumulative (the *Initial Non-Cumulative* flag is set in the DataSource).

   > ⓘ Note
   >
   > - For certain sources, only extraction mode *Full* is offered.
   > - For a CompositeProvider as a source, *Delta* mode is only available if the CompositeProvider is made up entirely of standard DataStore objects, Data Mart DataStore objects. A DataStore object can only be the target of a delta from a CompositeProvider if the CompositeProvider is made up entirely of DataStore objects.

General Extraction Settings - Request Selection

4. If you have extraction mode *Delta*, you can define further parameters:
   a. *Get all new data request by request*: Here you define how new data is retrieved from the source.

      Since a DTP bundles all transfer-relevant requests from the source, it can generate very large requests. If you do not want to use a single DTP request to transfer the dataset from the source because the dataset is too large for example, you can set the *Get all new data request by request* flag. This specifies that you want the DTP to read only one request from the source at a time. Once processing is completed, the DTP request checks for further new requests in the source. If it finds any, it automatically creates an additional DTP request.

> **Note**
>
> You can change this flag at any time, even if data has already been transferred. If you set this flag, you can transfer data request by request as a one-off activity. If you deselect the flag, the DTP reverts to transferring all new source requests at one time at periodically scheduled intervals.

b. *Only get delta once*: Here you specify whether the source requests should only be transferred once.

   Setting this flag ensures that the content of the InfoProvider is an exact representation of the source data.

   A scenario of this type might be required if you always want an InfoProvider to contain the most up-to-date data set for a source, but technical reasons prevent the DataSource on which it is based from delivering a delta (new, changed, or deleted data records). For this type of DataSource, the current data set for the required selection can only be transferred using a Full Update.

   In this case, a DataStore object cannot usually be used to determine the missing delta information (overwrite and creation of delta). If this is not logically possible because data is deleted in the source without delivering reverse records for example, you can set this flag and perform a *snapshot scenario*. Only the most up-to-date request for this DataSource is retained in the InfoProvider. Earlier requests for the DataSource are deleted from the (target) InfoProvider before a new one is requested (this is done by a process in a process chain, for example). They are not transferred again during the DTP delta process. When the system determines the delta when a new DTP request is generated, these earlier (source) requests are seen as *already fetched*.

c. *Perform delta initialization without data*: Here you specify whether the first request of a delta DTP should be transferred without any data.

   If this flag is set, the source data is flagged as fetched, but is not actually transferred to the target. The data flagged as fetched is not read by the next request. Instead only the new source data accrued in the meantime is read. Use this setting in scenarios where the DTP is automatically implemented within a process chain to rebuild a data target.

   > **Note**
   >
   > This setting is a transport-relevant change to the metadata of the DTP. If you want to flag source data as fetched just for test purposes, use the *Set delta status to 'Fetched'* option. You thus prevent a transport-relevant change being made to the metadata of the DTP.

d. *Extract all green requests*: This flag is displayed if the source of the data transfer process is a staging DataStore object that has only one inbound table. If you set this flag, all green requests are extracted, regardless of whether there are yellow or red requests between them. If the flag is not set, only those green requests are extracted that are older than the first yellow or red request.

5. *Only retrieve last request*: For a data transfer process that loads data in Full mode from a DataSource or an InfoPackage, you can stipulate here that the request of the data transfer process only retrieves the last request in the queue.

Execution

6. *Processing Mode*: The system normally defines the processing mode for the background processing of the respective data transfer process automatically.
7. *Technical Request Status*: Specify which technical status the request should be given if there are warnings in the log.
8. *Overall Status of Request*: Specify how the overall status of the request is defined.

Once the technical processing stage for a DTP request has been completed, the overall status of the request can be set automatically (based on the technical status of the request) or set manually (by the user). If the overall status is set manually, the status initially remains unchanged, if the technical processing stage was completed with a red or green status. In particular, this means that data for a green request is not released for reporting or further processing. The overall status has to be set manually by the user or by a process in a process chain.

9. *Automatically repeat red requests*: Specify whether red requests should be automatically repeated in process chains.

   If a DTP resulted in a canceled request during the previous run of a periodically scheduled process chain, this setting is evaluated the next time the process chain is started. If the flag is set, the previous request containing errors is automatically deleted and a new one is started. If the flag is not set, the DTP is terminated and an error message appears explaining that a new request cannot be started until the previous request is either repaired or deleted.

   > ⓘ Note
   >
   > If a delta update is carried out, the repaired request not only contains the data of the terminated request, but also all the data that has been added to the source since then. This can cause performance and consistency problems. Example of a consistency problem: A transfer with the prerequisite that data can only be transferred for one day at a time. This transformation then produces incorrect results, which leads to consistency problems if, during the repair, data for more than one day is to be transported at the same time.

4.16.3 Editing Extraction Properties

On the *Extraction* tab page, the extraction-specific properties of the source object are displayed. You can define and change certain extraction-specific properties for the source object, define filter and grouping for the extraction.

Procedure

Type of the source object

1. You can call the editor for the source object directly.

Settings for extraction from the source object

2. The system displays the extraction properties for the source object.
3. Depending on the source object, you can define and change settings.

Filter

4. If the source offers selectable fields, you can define filter criteria for data transfer.

   This means that you can use multiple data transfer processes with disjunctive selection conditions to efficiently transfer small sets of data from a source into one or more targets, instead of transferring large volumes of data. The filter thereby restricts the dataset to be transferred. You can specify single values, multiple selections, intervals, selections based on variables, or routines.

a. Using *Select*, you can select fields from the source for the filter.
b. To define a filter for a field, select the field and choose *Restrict*.
c. To create or edit a filter routine for a field, select the field and choose *Maintain*.

> **Note**
>
> You can now copy filter definitions for all objects by choosing *Copy Filter* and *Insert Filter*. When inserting a filter, the system checks whether it is suitable. If the system finds that this is not fully the case, it asks whether it should perform a comparison. In this case, the system only copies the filter for the fields found to be relevant.
>
> This works across project boundaries.
>
> Note however that this function only works within the Eclipse instance that is currently running. It cannot be called if routines would be affected.

Extraction Grouped by

5. Specify how you want to build the data packages that are read from the source (DataSource or InfoProvider).

   To do this, choose the fields of the source that you want to form a semantic key for the grouping. Data records that have the same key are combined in a single data package and processed together.

   With this setting, you also define the key for handling of data records with errors. You thus ensure that the data can be updated in the target in the correct order once the data records with errors have been corrected.

4.16.4 Editing Update Properties

On the *Update* tab page, properties for updating to the target object are displayed. You can set and change the properties for updating to the target object. You can also stipulate whether error handling is performed, and how.

Procedure

Target Object Type

1. You can call the editor for the target object.

Properties for Updating to the Target Object

2. The system displays the properties for updating to the target object.
3. Depending on the target object, you can make and change settings.

Error Handling

4. Make the settings for error handling.

   For more information, see Specifying How Data Records Are Handled [page 541].

4.16.4.1 Specifying How Data Records Are Handled

To write data records with errors to the data transfer intermediate storage, so that they can be updated to the target of the data transfer process once the error has been resolved, a number of settings have to be made in the data transfer process (DTP).

Context

In the default setting, error handling is completely deactivated (option *Request is canceled, records are not tracked and target is not updated*). This setting produces the best performance in all cases where the data quality is satisfactory, and the data flow has been tested. If the *Track records after failed request*field is selected on the *Update* tab, the DTP is executed again in the following cases with the option *Request is canceled, first incorrect record is tracked and target is not updated*:

- If a red request is restarted (because of the DTP setting *Automatically repeat red requests in process chains* for example).
- If the check triggered by the *Track records after failed request* field when request processing begins detects that the previous request has the technical status red.

If a request contains an error, selecting this field makes it possible to perform error analysis if executed again, and to track records back to the source in order to find out which one caused the error. Please note that the *Terminate request; First record with errors is traced; No updating* option and the *Activate Monitoring Automatically* checkbox are not available for SAP HANA execution.

We recommend activating error handling if errors occur during execution of the DTP. If error handling is activated, the data records with errors are written to the data transfer intermediate storage, where they can be corrected before being written to the data target using an error DTP.

Procedure

1. Define the key for error handling under *Extraction Grouped By* on the *Extraction* tab.

   If errors occur, all subsequent records with the same key are written to the data transfer intermediate storage along with the incorrect record; they are not updated to the target. This guarantees the serialization of the data records, and consistent data processing. The serialization of the data records and thus the explicit definition of key fields for the error stack is not relevant for targets that are not updated by overwriting.

   > ⓘ Note
   >
   > The key should be as detailed as possible. A maximum of 16 key fields is permitted. The fewer the number of key fields defined, the more records are updated to the data transfer intermediate storage.
   >
   > For targets that are not updated by overwriting, the system automatically creates the key fields of the target as key fields for error handling. In this case, you cannot change the key fields.

2. On the *Update* tab page under *Request Handling*, specify how you want the system to respond to data records with errors.

For error handling, the following options are available. The main difference between them is how the status of the request is set once all data packages have been processed:

- *Request is set to failed, error stack is written and valid records are updated*
- *Request is set to success, error stack is written and valid records are updated*

3. Specify the acceptable number of invalid records for error handling. If more than this specified number of errors occur, the transfer is terminated.

> **Note**
>
> If you leave this blank, handling for incorrect data records is not activated, and the update is terminated as soon as the first error occurs.

4. On the *Runtime Properties* tab, make the settings for the temporary storage .

   The intermediate storage supports you with troubleshooting and when restarting the load process. In these settings, you specify the processing steps after which you want the system to temporarily store the DTP request (such as extraction, filtering, removing new records with the same key and transformation). You also specify when the temporary storage should be deleted. This can be done either after the request has been updated successfully to the target, when the request is deleted or a specified amount of time after the request is processed. Under *Level of Detail*, you specify how you want to track the transformation.

5. Once you have activated your data transfer process, create an error DTP on the *Update* tab page by pressing  and include it in a process chain or start manually if errors occur.

6. Analyze the error and correct the data records with errors.

   The intermediate storage for the individual processing steps can be displayed in the Request Monitor for the DTP by pressing the button in the *Data* column. The records with errors can be identified from the icon.

   If the request contains records with errors, you can call the data transfer intermediate storage (DTIS) (error stack) by pressing *Open DTIS Maintenance* on the *Update* tab in the data transfer process and correcting the records there. The authorizations for DTIS maintenance are checked using authorization object S_RS_DTP. You require activity 23 (Maintain DTP Definition).

7. Execute the error DTP manually or via a process chain.

4.16.4.2 Options for Error Handling

You have the following options in the data transfer process maintenance transaction on the *Update* tab under *Error Handling*:

Options for Error Handling	Error Handling is....	Explanation
Request is canceled, records are not tracked and target is not updated	Fully deactivated.	This is the default setting in the data transfer process. With the first error, updating is terminated for the entire data package, and the error is reported as a package error in the DTP monitor. The request is not released for reporting. The incorrect records are not written to the data transfer intermediate storage (error stack), since the request has been terminated and has to be completely updated again. The error is not assigned to the data record. The cross-reference table to determine the data record number is not created. This allows for faster processing. This option produces the best performance in all cases where the data quality is satisfactory, and the data flow has been tested. With this option, selecting the *Track records after failed request* field provides you with the option of switching error handling to *Request is canceled, first incorrect record is tracked and target is not updated* in the following cases: • If a red request is restarted (because of the DTP setting *Automatically repeat red requests* for example). • If the check triggered by the *Track records after failed request* field when request processing begins detects that the previous request has the technical status red. With this setting, the cross-reference table is created, thus allowing better troubleshooting. If errors occur, the cross-reference table makes it possible to trace back to the source in order to find out which data record caused the error.

Options for Error Handling	Error Handling is....	Explanation
Request is canceled, first incorrect record is tracked and target is not updated	Deactivated.	The system behaves the same as with the *Request is canceled, records are not tracked and target is not updated* option, with the difference that the cross-reference table for determining the data record number is created during the processing of all DTP data packages. If errors occur, the cross-reference table makes it possible to trace back to the source in order to find out which data record caused the error.
*Request is set to failed, error stack is written and valid records are updated*Request red; Write error stack; Update valid records	Activated.	If errors occur, the technical status of the request is set to red once the data packages have been processed. This option makes it possible to update valid data that is been released for reporting. This does not happen however until after the updated records have been checked, and the request has been released manually by setting the overall status. The records with errors are written to a data transfer intermediate storage where they are edited and can be updated manually using an error DTP.
*Request is set to success, error stack is written and valid records are updated*Request green; Write error stack; Update valid records	Activated.	If errors occur, the technical status of the request is set to green once the data packages have been processed. The valid records are available immediately for reporting and analysis purposes. Any subsequent actions are performed automatically. The records with errors are written to a data transfer intermediate storage where they are edited and can be updated manually using an error DTP.

4.16.5 Editing Runtime Properties

On the *Runtime Properties* tab, make the settings for the temporary storage and BW background processing . You can only make settings for temporary storage if processing is not in SAP HANA.

Context

Temporary storages are tables that contain the data records that have been processed in a particular step during the data transfer process request processing. They are used at runtime to allow processes to be restarted without a significant impact on performance and consistency when errors occur.

BW Background processing enables you make load processes parallel.

Procedure

Temporary Storage

1. *Level of Detail*: Specify the level of detail.
2. *Fill Storage After Selected Steps*: Specify which request processing steps temporary storage should be written for.
3. *Delete Storage*: Specify when the temporary storages should be deleted again.

Background Processing

4. *Number of Parallel Processes*: Define the maximum number of work processes that should be used to process the data transfer process.
5. *Job Class of Processes*: Specify a job class for defining the job priority.
6. *Servers*: Define which server(s) to execute the work processes process on.

4.16.6 Excuting Data Transfer Process

Execution of the data transfer process is generally managed using a process chain. In the data transfer process editor, you can execute a data transfer process directly in the background, simulate execution in debugging mode, or execute a delta without transferring any data for test purposes.

With the *Start execution of data transfer process* function, you have the following options:

- *Execute in Background*: The data transfer process is started directly in the background. You can stipulate taht the monitor is called after execution.
- *Simulate in Dialog*: If you want to execute the data transfer process in debugging mode, choose this option. The system now displays the program flow. You can set breakpoints here. The request is processed synchronously in a dialog process, and the update of the data is simulated. If you select Expert Mode, you can also define selections for the simulation and activate or deactivate intermediate storage. You can set breakpoints as well. More information: Simulating and Debugging DTP Requests [page 546].
- *Set delta status to 'Fetched'*: If you want to execute a delta without transferring any data for test purposes, select this option. This option is available when the data transfer process extracts in delta mode. In this case you execute the DTP directly in the dialog. A request that is started in this way marks the data that is found in the source as fetched, without actually transferring it to the target. You can also use this option if delta requests have already been transferred for this data transfer process. This processing mode implies that no transport-relevant changes have been made to the DTP metadata.

4.16.6.1 Simulating and Debugging DTP Requests

Simulating the data update and debugging a data transfer process request (DTP request) help you to analyze an incorrect DTP request. You can also simulate a transformation prior to the actual data transfer if you want to check whether it provides the desired results.

> **Note**
>
> Requests that are processed with processing mode SAP HANA Execution cannot be debugged.

Procedure

Simple Simulation and Debugging

1. If you want to execute a data transfer process in debugging mode, choose function *Start execution of data transfer process* > *Simulate in Dialog* in the data transfer process editor.
2. On the next screen, in the overview for program execution, you define breakpoints for debugging by choosing *Change Breakpoints*.
3. Choose *Simulate*.
   The simulation request is started. The request is processed synchronously in a dialog process. If you have defined breakpoints, the ABAP Debugger now opens. If you have defined a breakpoint for the transformation, you can call the corresponding data in the ABAP source code from the *Data* section.
   The result of the simulation is displayed in the DTP monitor.

Simulation and Debugging in Expert Mode

In addition to setting breakpoints, expert mode enables you to specify selections for simulating the request and for defining the substep after which you want to write to temporary store during the simulation .

1. You can perform the simulation of a request in expert mode from the data transfer process editor or from the DTP monitor.
   1. In the data transfer process monitor, select the execution option *Simulate in dialog*. On the next screen, select the *Expert Mode* field and choose *Simulate*.
   2. In the DTP monitor, choose *Debugging*.

   The *Debug Request* screen appears.
2. Make the settings on the *Filter* tab page.
   When you call the simulation from the DTP monitor, the filter settings that are displayed as initial are copied from the request displayed in the monitor. When you call from the data transfer process monitor, the filter settings displayed as initial are copied from the next "real" request of the DTP.
   - You can change the filter in order to restrict the data for the analysis. In addition to selection via the fields of the source, you can also select via the the technical fields of the request . You should restrict the data in order to prevent the simulation from running for a long time, for example, if you loaded a large amount of data. Selections are also recommended if the fields refer to certain data records in the source or to a certain selection of data in the source.
   - With *Save as Variant*, you can store the selections in the database for later use.
   - With *Load Variant*, you can call a selection variant stored for this request.

- With *Technical Field Name / Description*, you can switch between displaying the technical field name and the description.
- With *Change Selection*, you can change the list of fields displayed in the selection screen.

3. Make your settings on the *Temporary Storage and Breakpoints* tab page.
   You can define if and when to write data to temporary storage during processing and set breakpoints at different times.
4. Choose *Execute*.
   The simulation request is started. The request is processed synchronously in a dialog process. If you have defined breakpoints, the ABAP Debugger now opens. If you have defined a breakpoint for the transformation, you can call the corresponding data in the ABAP source code from the *Data* section.
   The result of the simulation is displayed in the monitor and can be analyzed.

4.16.7 Monitoring Data Transfer Process

A request is generated to execute a data transfer process (DTP). The *DTP Monitor* app allows you to keep track of the request in the various processing steps.

Prerequisites

The SAP BW/4HANA Cockpit has been configured. More Information:

Call Monitor

You can call DTP Monitor in the following ways:

- By choosing *Show Monitor* > *Show Last Request* from the data transfer process editor in the BW Modeling tools.
- By pressing the *DTP Monitor* button in the SAP BW/4HANA Cockpit in the request administration screen for a DataStore object or InfoObject in the details for the load request.
- Using the *Process Monitor* (*Processes* app).

Display Header Information for Request

You can use the arrow icon to display the header information for the request. This provides detailed information about the request, together with various navigation options. You can navigate to the source object, to the data transfer process or to the transformation for example if you have already opened the BW Modeling tools, or you can navigate to the administration screen for the the data target. For a streaming request, you can navigate to the previous and subsequent request, and display the streaming process.

Display Detailed Information About the Processing of the Request

The Request Monitor for a request in the data transfer process request displays detailed information about the processing of the request. In the *Request Processing Steps* column, you can see the request processing steps in a hierarchical tree display. At the first level of the detail tree, you can see the status of the requests at various points during the process, for example during generation, when processing the steps during the DTP program, and when setting the technical and complete status of the request. The status display for the processing is then divided further according to the steps of the data transfer process and displays the status and messages for each of the processing steps and data packages in the processing steps.

In the *Details* column, you can call notifications or other information for the entries in the Request Monitor, or navigate to the transformation that was executed in this step. At the processing step level, you can call the temporary storage display for this processing step for example if you have specified for the data transfer process that the temporary storage is to be filled after this processing step. At the transform execution level, you can navigate to the transform for example if you have opened the BW Modeling tools.

The buttons at the top right of the screen provide you with various navigation options. You can call the job overview, the request administration screen for the data target, and the DTP process monitor, where you can see an overview of all requests belonging to the DTP. You can call to error stack, if one exists, or, if the DTP is executed via a process chain, call the process chain logs for the process chain run that the request was executed with. You can also switch to another request, using the TSN, request ID or SID.

The toolbar at the bottom right of the screen provides you with the following features:

- *Delete*
  You can delete a request if it has not been updated to a target yet.
- *Repair*
  You can repair a request that has errors. The repair can be performed either synchronously or asynchronously. When a request is repaired, a new run ID is created, and the system attempts to write the data package that had errors again. In the *Details* section in the Request Monitor, the processing steps that are repeated are displayed with the run ID in brackets.
- *Change Overall Status*
  You can specify the overall status of the request here, regardless of its technical status.

4.16.8 Displaying Properties of the Data Transfer Process

The *Properties* view displays information about the data transfer process (DTP) in the active editor.

General

Administrative information is displayed here, such as the assigned package, the version, information about the person responsible or the last change.

Technical

The DTP ID and the DTP type are displayed here, as well as information about the error DTP and generated programs and objects.

Usage

The data flow for the DTP (source, destination, transformation) is displayed here, together with the process chains if the DTP is used in process chains. You can navigate from here to the data flow display or the process chain display transaction.

Program Flow

The program flow for the data transfer process is illustrated here.

5 Security Aspects for the Modeling Tools in Eclipse

The following sections explain security aspects that you should bear in mind when using modeling tools for SAP BW/4HANA.

Target Group

- System administrators

Why is Security Important?

With the increasing use of distributed systems and the Internet for managing business data, security requirements are also becoming more prominent. When using a distributed system, you need to be sure that your data and processes support your business needs without allowing unauthorized access to critical information. It is very important that user errors, negligence, or attempted manipulation do not result in loss of information or affect processing time. These security requirements also apply to Eclipse modeling tools. We have provided this information to help you to make modeling tools more secure.

Related Information

User Authentication on the Front End Client [page 550]
Authorizations for Modeling with the Modeling Tools [page 551]
Protecting Resources on the Front End Client [page 553]
Installing Third-Party Plug-Ins [page 554]

5.1 User Authentication on the Front End Client

In modeling tools, you always work with BW projects in order to access metadata objects from the back end system (SAP BW/4HANA).

A BW project represents a real system connection on the front end client. It therefore requires an authorized user in order to access the back end system. With the standard authentication method, the user enters a user name and password to log on to the back end system.

Risks

Standard authentication with explicit specification of a user name and password means that the user data entered on the front end client is loaded as plain text into the memory of the local host. A password that is saved locally is a potential security breach, as it could be extracted from the memory by third parties.

Security Measures

Activating **Secure Network Communication (SNC)** for the selected system connection is mandatory due to security reasons.

Use **Single Sign-On (SSO)** as well. When used with SNC, SSO also meets the security requirements for working with large-scale BW projects. With SSO, the user does not need to enter a user name and password. S/he can simply access the system as soon as the logon ticket has been checked.

> ⓘ Note
>
> Note that configuring SSO is a general configuration step. There is no difference to the configuration for ABAP Development Tools. If you have already configured SSO for ABAP Development Tools in your landscape, no further configuration is required for modeling tools.

Besides issuing logon tickets, AS ABAP systems can also issue restrictive assertion tickets when system services are accessed. If you use integrated SAP GUI applications in modeling tools, the assertion tickets provide a greater level of security. The back end system does not request a password. Instead it checks the validity of the assertion ticket to permit the user to access system services. We therefore recommend configuring your AS ABAP system to only issue assertion tickets.

More Information

For more information, see the installation guide for BW modeling tools.

5.2 Authorizations for Modeling with the Modeling Tools

Authorizations are assigned to users in the back end system. This assignment is based on roles that are predefined in the system. One or more roles are assigned to a user. These roles are based on authorization objects from a technical viewpoint.

Basis Authorizations

Standard role

The following role is required to use the modeling tools. If role SAP_BC_DBW_ABAPDEVELOPER is already assigned to a user, the role already contains the required authorizations.

Role	Description
SAP_BC_DWB_WBDISPLAY	Roll that contains all authorizations for displaying and searching ABAP development objects. **Note** The users are not allowed to modify ABAP development objects.

Authorization object S_RFC

The modeling tools require remote access to the following function modules that are specified for authorization object S_RFC:

Activity [ACTVT]	Name of RFC Object [RFC_NAME]	RFC Type [RFC_TYPE]
16 (Execute)	DDIF_FIELDINFO_GET RFCPING RFC_GET_FUNCTION_INTERFACE SADT_REST_RFC_ENDPOINT SUSR_USER_CHANGE_PASS-WORD_RFC SYSTEM_RESET_RFC_SERVER	FUNC (Function Module)

Authorization object S_TCODE

The modeling tools need to start specific transactions for SAP GUI integration in Eclipse. The BW modeling tools therefore need access to the following transactions, which are specified in authorization object S_TCODE:

- SADT_START_TCOD
- SADT_START_WB_URI

For more information, read the document Configuring the ABAP Back End for ABAP Development Tools in the SAP Community Network.

SAP BW/4HANA-Specific Roles and Authorizations

When working with BW modeling tools, you can only see or open objects that you have at least display authorization for. The same checks are performed for actions on objects in the modeling tools as for actions in

the back end system or in the query. We therefore recommend the following role templates for users who work with the modeling tools:

Role Template	Description
S_RS_RDEMO	BW role: modeler (development system)

> **Note**
>
> Notes on specific object types:
>
> If the authorization object has subobject field defined for an object type (TLOGO), the user needs to have authorization * or at least *Definition*, in order to see the object in the Project Explorer tree.

In particular, modelers need authorizations that are specified in the following authorization objects:

Authorization Object	Description
S_RS_HCPR (SAP HANA CompositeProvider authorizations)	Authorizations for working with CompositeProviders and their subobjects
S_RS_ODSV (Open ODS view)	Authorizations for working with Open ODS Views
S_ADT_RES	Authorization object for ADT resource access Authorization field URI must have the value */sap/bw/modeling/**. **Note** The placeholder "*" is used for the URI subfolders.

5.3 Protecting Resources on the Front End Client

In the modeling tools, a BW project represents a user-specific view of the BW metadata objects of the back end systems (SAP BW/4HANA).

Like all projects in Eclipse, BW projects also have a local representation of their data on the front end and are managed in a workspace. If you have a BW project, there will therefore be local copies of the SAP BW/4HANA metadata objects on the front end. This means that it is possible to access metadata located outside of the SAP repository at local file system level.

Risks

The SAP BW/4HANA metadata objects can be found by third parties.

Security Measures

To protect local project resources, we recommend creating workspace folders to store project resources locally, in order to prevent third parties from accessing the resources. Use the existing security measures that are available at operating system level.

> ⓘ Note
>
> Files stored under Windows in the personal substructure of a user can only be accessed by that user or by local administrators.

> → Tip
>
> We especially recommend using the default workspace that was created when the integrated development environment (IDE) was installed.

5.4 Installing Third-Party Plug-Ins

Your installation of modeling tools can be enhanced by using additional plug-ins from various third-party providers.

Risks

These plug-ins can take control of your client installation or even take control of your complete front end PC.

Security Measures

Be very careful when deciding which plug-ins to install.

6 What's New in the BW Modeling Tools

The table below provides an overview of the versions of the BW modeling tools together with the relevant release information notes. These describe any dependencies on ABAP development tools, SAP HANA Studio and SAP NetWeaver Support Package Stacks.

BW Modeling Tools	Release Information Note (RIN)
Version 1.23	Patch 0: 2979762
Version 1.22	Patch 0: 2866670
Version 1.21	Patch 0: 2738680 Patch 1: 2759015 Patch 2: 2772316 Patch 3: 2791573 Patch 4: 2835115
Version 1.19	Patch 0: 2597817 Patch 1: SAP Note 2621659 Patch 2: SAP Note 2624373 Patch 3: SAP Note 2662821 Patch 4: SAP Note 2695308 Patch 5: SAP Note 2717030
Version 1.18	Patch 0: SAP Note 2449272 Patch 1: SAP Note 2471954 Patch 2: SAP Note 2481081 Patch 3: SAP Note 2487263 Patch 4: SAP Note 2505002 Patch 5: SAP Note 2537282
Version 1.17	Patch 0: SAP Note 2407411 Patch 1: SAP Note 2420864 Patch 2: SAP Note 2432650

BW Modeling Tools	Release Information Note (RIN)
Version 1.16	Patch 0: SAP Note 2371987 Patch 1: SAP Note 2393196 Patch 2: SAP Note 2401857
Version 1.15	Patch 0: SAP Note 2349841 Patch 1: SAP Note 2361868 Patch 2: SAP Note 2375071
Version 1.14	Patch 0: SAP Note 2326527
Version 1.13	Patch 0: SAP Note 2246221 Patch 1: SAP Note 2284433 Patch 2: SAP Note 2303903
Version 1.12	Patch 0: SAP Note 2233558 Patch 1: SAP Note 2244647
Version 1.11	Patch 0: SAP Note 2202179 Patch 1: SAP Note 2228822
Version 1.8	Patch 0: SAP Note 2164033 Patch 1: SAP Note 2175822 Patch 2: SAP Note 2175886
Version 1.7	Patch 0: SAP Note 2067815 Patch 1: SAP Note 2138863 Patch 2: SAP Note 2149780
Version 1.6	Patch 0: SAP Note 2067814 Patch 1: SAP Note 2104488 Patch 2: SAP Note 2108620
Version 1.5	SAP Note 2067813
Version 1.4	SAP Note 2030817

6.1 Version 1.25

This section describes all new features, enhancements and changes in connection with BW Modeling Tools in version 1.25.

New back end-specific features are available in back-end version SAP BW/4HANA 2021 SPS 04.

6.1.1 Merge Mode for DataStore Objects (New)

In the BW Modeling tools, you can now change the mode for execution of the delta merge in the technical settings for DataStore objects.

- *Smart Merge*: This is the default setting for all DataStore objects with the exception of DataStore objects that have a write interface. Whenever changes are made to the data of a DataStore object, the system triggers a *smart merge*.
- *Auto Merge*: This is the default setting for DataStore objects that have a write interface. Whenever changes are made to the data of a DataStore object as part of the write process, the system does **not** trigger a *smart merge* because a delta merge is executed for the activation in any case. This option is particularly useful in streaming and push scenarios and is therefore preset as the default setting for DataStore objects that have write interfaces.

> ⓘ Note
>
> More information: DataStore Object with Write Interface [page 79] and .

Related Information

Technical Settings [page 286]

6.1.2 Mass Activation (New)

In the *BW Modeling tools*, you can use the *mass activation* view to create a collection of SAP BW/4HANA objects, collect their dependent objects, and trigger activation of the selected objects.

6.1.3 Transport Overview (New)

In the *BW Modeling tools*, the *BW Transport Overview* provides you with an overview of your modifiable transport requests and allows you to add more SAP BW/4HANA objects.

6.1.4 Enhanced Master Data Update - Mixed Case (Enhanced)

For characteristics with attributes and/or texts, you can use enhanced master data update. This enables parallel loading of data. You can also use now enhanced master data update for characteristics that have both time-independent and time-dependent attributes (mixed case).

Related Information

Master Data Update [page 58]

6.1.5 InfoObject with Write Interface (New)

It has been possible for some time now to use DataStore objects with a write interface. This property is now also available for InfoObjects. Texts and attributes are supported by the familiar master data types. The data in these InfoObjects can be moved to the inbound table using a tool such as Data Services or via SAP Cloud Platform Integration (CPI) by PUSH.

As a prerequisite for this you must be using the enhanced master data update (see Master Data Update [page 58]).

> **Note**
>
> Note that XXL attributes and hierarchies are **not** supported.

To set this property, go to the corresponding InfoObject in the BW Modeling tools and choose *Properties* > *Enhanced Master Data Update* > *Write Interface-Enabled*.

You can find the URIs specific to each InfoObject to call the functions using HTTP (JSON or XML) in the BW Modeling tools, under *Properties* > *Template URIs*. There are specific URIs for attributes and texts.

The RFC function modules are contained in function group RSO_PUSH. The same function modules are used as for DataStore objects with write interface, with the addition of object_type = IOBJ and object_subtype = ATTR or TEXT.

Related Information

InfoObject with Write Interface [page 59]

6.2 Version 1.23

This section describes all new functions, enhancements and changes in connection with BW Modeling Tools in version 1.23.

New back end-specific functions are available in back-end version SAP BW/4HANA 2.0 SPS 07.

6.2.1 Content Explorer View and Content Install View (New)

Preconfigured BW objects are delivered in SAP BW/4HANA Content as part of the ready-made information models for analyzing business issues. The following sections explain how you can use the *BW Modeling tools* to transfer SAP BW/4HANA content objects to the active version for productive use.

6.2.2 Operations in SAP HANA (Enhanced)

In the documentation for this query runtime profile property, you can find information about the new options for calculating exception aggregation in SAP HANA.

Fewer restrictions apply now when using variables with processing type “Replacement Path”, with replacement from the attribute value of a characteristic.

Related Information

Exception Aggregation in SAP HANA [page 425]

6.2.3 DataStore Object - Dynamic Partitioning (Enhanced)

Dynamic partitioning is more flexible than static partitioning.

For dynamic partitioning, you only specify the partition field. The partitions are then created dynamically during activation of the data. The new feature provides automatic deletion of partitions if they do not contain any data during activation of the data.

With this SP, you can flexibly change the parameters for dynamic partitioning if the DataStore object contains data.

You can change the partitioning characteristic, the granularity, or both at the same time.

You can also switch between dynamic partitioning and static partitioning.

Related Information

Creating Partitions [page 278]

6.2.4 DataStore Object - Technical Settings (Enhanced)

The expert settings for the technical properties of a DataStore object have been enhanced.

As of SP 07, you can specify the expected number of data records for an InfoProvider. This information is used during creation and activation of a (empty) DataStore object in order to create an appropriate number of hash buckets on the SAP HANA database.

Related Information

Technical Settings [page 286]

6.2.5 Data Tiering Optimization - Exceptional Updates (New)

You can update the data stored in an external cold store at regular defined time intervals. This should only be an exception however. You can also schedule these updates in process chains.

Related Information

Data Tiering Properties [page 81]

6.3 Version 1.22

This section describes all new functions, enhancements and changes in connection with BW Modeling Tools in version 1.22.

New back-end-specific functions are available in back end version SAP BW/4HANA 2.0 SPS 04.

6.3.1 Query: Restrict Formula (New)

In a key figure structure, you can restrict a local formula in a further selection.

The restricted selection can only be used in the context of the current query.

Related Information

Restrict Formula [page 343]

6.3.2 Query: Precalculate Value Set (New)

A precalculated value set can be used as a filter in an analytic query.

The filter is defined as a variable of type *predefined value set* and is executed at runtime. If the precalculated value set is large, query runtime can be improved considerably. A precalculated value set is always assigned to a characteristic that is used in the query.

Related Information

Precalculating Value Sets [page 375]
Filling a Value Set [page 376]
Defining a Variable for a Precalculated Value Set [page 377]
Using Precalculated Value Sets [page 378]

6.4 Version 1.21

This section describes all new functions, enhancements and changes in connection with BW Modeling Tools.

New back-end-specific functions are available in back end version SAP BW/4HANA 2.0 SPS 0.

6.4.1 DataStore Object (Changed)

In the editor for creating and changing DataStore objects in the BW Modeling tools, there are a number of changes in the UI and some functional enhancements:

- The modeling properties of DataStore objects have been completely revised.
- In the modeling properties of DataStore objects, you can use the *Write Interface enabled* property. The data in this DataStore object can be moved to the inbound table using a tool such as Data Services or via SAP Cloud Platform Integration (CPI) by PUSH.
- In the settings for Data Tiering Optimization, you can specify a cold store connection.
- When editing the fields of the DataStore object, you can remodel fields and InfoObjects.

Related Information

Modeling Properties [page 71]
DataStore Object with Write Interface [page 79]

6.4.2 External SAP HANA SQL View for DataStore Objects (New)

The external SAP HANASQL view is a stable SAP HANA view which you can use to access the data in a DataStore object directly.

Related Information

External SAP HANA SQL View [page 88]

6.4.3 InfoObject (Enhanced)

In the Editor for creating and changing InfoObjects in the BW modeling tools, you can make settings for read access logging (RAL).

Related Information

Editing Various Settings [page 264]

Creating Key Figures [page 265]

6.4.4 Data Flow Modeling (Enhanced)

A number of new functions have been added to data flow modeling in the BW Modeling tools.

- You can now display HTML documentation for the data flow as an HTML file and save it locally on your computer or print/save is as a PDF file.
- In a data flow containing a CompositeProvider with SAP HANA views as participating InfoProviders, the system now also displays the SAP HANA views. Previously this was not possible. If the SAP HANA system is linked with the BW project, you can open the SAP HANA view from the Data Flow Editor.
- Usability has been improved for opening transformations and data transfer processes (DTPs). You can now open the Transformation Editor or DTP Editor by double-clicking on the corresponding icon in the InfoProvider or InfoSource. The context menus for transformation or DTP also now provide the relevant functions.
- On the *General* tab, you can specify that all transformations and data transfer processes that previously were not included in the data flow object, and whose source and target object are part of the data flow object, are now included in the data flow object. Once you have made the setting and have saved and activated the data flow, the transformations and data transfer processes are displayed in the editor.

Related Information

Saving Data Flow Documentation as an HTML File [page 209]
Working with the Data Flow Object Editor [page 198]
Functions in the Data Flow Object [page 201]

6.4.5 Enhanced Data Flow Copy (New)

You can use the enhanced data flow copy to create one or more new data flows from an existing one and thus to swap, recreate or remove the objects in the target data flow.

You can use an existing data flow as a template for other data flows for example or copy a data flow from SAP BW/4HANA Content and then adapt it to your requirements.

Related Information

Copying a Data Flow [page 210]

6.4.6 Grouping of Process Chains into InfoAreas (New)

Previously, the system used display components to group process chains. In the SAP BW/4HANA cockpit, the application components were called groups. To allow the integration of process chains into the BW Modeling tools like other SAP BW/4HANA metadata objects, they are now grouped into InfoAreas.

Process chains are now displayed in the InfoAreass in the *BW Repository* tree and can be created and opened using the BW Modeling tools. General functions such as the BW object search or the Open dialog and context menu functions in the *BW Repository* tree, such as the where-used list or deleting process chains, as supported for process chains.

During the upgrade, existing display components or groups are converted into InfoAreas. For more information, see *SAP Note 2732179*.

Related Information

InfoArea [page 40]
SAP Note 2732179

6.4.7 Variable with Replacement from BRFplus (New)

Variables of processing type *BRFplus Exit* make it possible to access the functions of the SAP rule engine *Business Rules Framework plus* (BRFplus) in SAP BW/4HANA.

In the *BW Modeling Tools*, you assign a a BRFplus function to a variable of this processing type.

Related Information

BRFplus Exit [page 136]
Using a Variable with Replacement from BRFplus [page 392]

6.4.8 Operations in the SAP HANA (Enhanced)

The system support calculation of conditions in the SAP HANA (expert mode *Conditions Calculated in SAP HANA (9)*).

In the default mode *Offensive (P)*, the optimizer for operations in SAP HANA decides whether the query is executed with expert mode *Conditions Calculated in SAP HANA (9)* or expert mode, *Optimized Access (3)*.

Related Information

Operations in SAP HANA [page 419]
Conditions Calculated in the SAP HANA [page 433]

6.4.9 Currency Translation and Unit Conversion (Changed)

Editors are now provided for currency translation and unit conversion in the BW Modeling tools.

6.4.10 Key Date Derivation Type (Changed)

For the key date derivation type, there is an editor in the BW Modeling tools. The editor can be called from the context menu of any InfoArea. The key date derivation type is then assigned to central InfoArea `0BW4_GLOBAL` however.

6.4.11 BW Job Repository Feed (New)

In the *Feed Reader* in the BW modeling tools, you can now subscribe to the *BW Job Repository* feed.

If you have subscribed to the feed for a system, the *Feed Reader* view will display the following system jobs in accordance with the filter criteria set in the feed query:

- DATAFLOWCOPY
- REMODELING
- DTP_LOAD (DTP execution using the BW Modeling tools)
- DTO

Related Information

Subscribing to the BW Job Repository Feed [page 22]

6.4.12 BW Job View (New)

The *BW Job view* in the BW Modeling tools provides you with an overview of the BW jobs in your BW projects, together with details about the jobs and job logs. You can use filters to specify which jobs are displayed. If the

job type supports this, you can perform activities from the view, such as interrupting a job that is running or restarting a failed job.

Related Information

BW Job View [page 17]

6.4.13 BW Modeling Tools - Miscellaneous (Changed)

The following changes have been made:

- The specification of an object type in the object search and in the Open dialog has been revised.
- You can reset an object that has been changed and saved back to the previous active version. The modified version (M version) is thereby deleted. This function is available for InfoObjects, DataStore Objects and CompositeProviders.
- In the BW Repository tree, you can now find InfoArea `0BW4_GLOBAL` - *Global objects*. This contains the following objects:
  - Currency translation types
  - Key date derivation types
  - Quantity conversion types
  - Variables

Related Information

Searching for and Opening Objects with the Open BW Object Dialog Box [page 24]
Searching for SAP BW/4HANA Objects [page 25]
Reset SAP BW/4HANA Object to Active Version [page 33]
InfoArea [page 40]

6.5 Version 1.19

This section describes all new functions, enhancements and changes in connection with BW Modeling Tools. New back-end-specific functions are available in back end version SAP BW/4HANA 1.0 SPS 08.

6.5.1 Transformation (changed)

You can now create and edit transformations in the BW modeling tools. The transformation features in the Data Warehousing Workbench have been transferred to the editor in the BW modeling tools.

To edit ABAP or SAP HANA script routines in the BW modeling tools, you require a development key and an ADT Eclipse connection to the SAP BW/4HANA-System.

Transformations can no longer be created and edited in the Data Warehousing Workbench.

The following new features are provided:

- The runtime of the transformation can be switched from SAP HANA to ABAP and vice versa
- ABAP end routines can be created in the SAP HANA.runtime
- The SAP HANA runtime now supports all possible source object types
- Semantic grouping is supported for the SAP HANA runtime
- Mass maintenance of the aggregation type
- Currency translation with constant
- Editing the error stack

Related Information

Transformation [page 187]
Creating Transformations [page 497]

6.5.2 Data Transfer Process (Changed)

You can now create and edit transformations in the BW modeling tools. The transformation features in the Data Warehousing Workbench have been transferred to the editor in the BW modeling tools.

Transformations can no longer be created and edited in the Data Warehousing Workbench.

The following changes and new features have been introduced in the data transfer process (DTP):

- The runtime of the data transfer process can no longer be set in the data transfer process. Instead, the system determines whether execution in SAP HANA is possible, or must be performed in ABAP, and creates the processing type accordingly. If possible, the data transfer process is executed in SAP HANA.
- For a data transfer process that is executed in SAP HANA, you can now also select source fields that form a semantic key for grouping. All records for the specified keys are then combined into one data package and processed together.
- If you have activated error handling, and data records with errors have been written to the data set intermediate storage (DTIS) as the error stack, you can call DTIS maintenance from the data transfer process editor in order to correct the records there.

Related Information

Data Transfer Process [page 190]
Creating a Data Transfer Process [page 535]

6.5.3 Optimizer for Operations in SAP HANA (New)

The optimizer helps you to select the relevant mode in the system that can execute operations to your advantage in SAP HANA.

If you execute a query with a standard mode, the optimizer decides whether the query is executed with the mode *Optimized Access (3)* (negative decision) or using the enhanced expert mode of the standard mode (positive decision) is executed.

When importing SAP BW/4HANA 1.0 SPS08, note that the system executes a **one-time** automatic mapping of the previous (fixed) modes to the new (relative) standard modes:

Mode of Existing Query	Mapping to New Standard Mode
No optimized operations in SAP HANA (0)	*Defensive (J)*:
Individual Access per InfoProvider(2):	
Optimized Access (3):	
Exception aggregation in the SAP HANA (6)	*Standard (M)*:
Formulas Calculated in SAP HANA (7):	
Formulas Calculated in SAP HANA with Complex Currency or Unit of Measure (8)	

Experts also have the option to select a (fixed) expert mode. A prerequisite, however, is that they have set the appropriate switch. If a user activates the switch for expert modes and sets an expert mode for a query, this mode is retained.

> **Note**
>
> To activate, see 2505732
>
> Note that the optimizer for operations in the SAP HANAdatabase is **not** called when you select one of these expert modes.
>
> Also note the following case: A user activates the switch for expert modes and sets an expert mode for a query. After this, other users who use this query can also see the selected expert mode even if they have not activated the switch for expert modes.

Related Information

Operations in SAP HANA [page 419]
Optimizer for Operations in SAP HANA [page 421]
Learning Program for the Optimization of Operations in SAP HANA [page 422]

6.5.4 Editor for Document Store (New)

In suitable front ends, you now have the option in queries of documenting cells with numbers and storing these (text) documents in the back end. The documents are saved in the new TLOGO object saved `Document Store`.

For a query to be able to process documents, the new query runtime properties *Document Support* and *Default Document Application ID* have to be adjusted.

Note the authorization concept: In the query, you can document what you can see on the basis of the analysis authorizations for the query. To be able to with the document store and the documents contained in it, you also require the authorizations for standard authorization object `S_RS_DOCA`.

Related Information

Document in the Document Store [page 110]
Creating a Document Store [page 314]
Document Support [page 443]

6.5.5 Open ODS View-Wizard and Editor (Changed)

The wizard for creating Open ODS Views has been revised and given a more user-friendly design.

The wizard has been revised, incorporating the following changes:

- The initial screen of the wizard has been split into two separate screens: Selection of the source type has been moved from the first screen in the wizard to a second screen.
- You can now reference each wizard page by double-clicking on the next page.
- If after selecting a source type you navigate between the wizard pages, previous selections made in this scenario remain and are not lost.
- If you create a source system of type *SAP HANA Smart Data Access*, the system now selects the path prefix using a dropdown list instead of a slider.

The following change has been made in the editor:

- **Create Open ODS View Copy via Save As**
  By choosing *File* > *Save As*, you can copy an Open ODS view.

Related Information

Creating an Open ODS View Using the Wizard (Defining Semantics and Source) [page 216]

6.5.6 Wizard for Creating Source Systems (Changed)

The wizard for creating source systems has been revised and given a more user-friendly design.

The revision has the following changes:

- The initial screen of the wizard has been split into two separate screens: Selection of the connection type has been moved from the first screen in the wizard to a second screen.
- You can now reference each screen in the wizard by double-clicking on the next screen.
- If you navigate between the wizard screens after selecting a connection type, any prior selections made in this scenario are retained.

Related Information

Creating a Source System Using the Wizard [page 460]

6.5.7 Editor for File DataSource (New)

You can now create and edit DataSources for file source systems in the BW modeling tools. The features for editing file DataSources in the Data Warehousing Workbench have thus been transferred to the editor in the BW modeling tools.

Related Information

Creating DataSources for File Source Systems [page 488]

6.5.8 DataSource-Editor (Enhanced)

A number of new functions have been added to the editor for DataSources.

The following changes and enhancements are available:

- **Create a DataSource Copy by Choosing Save As**
  By choosing *File* > *Save As*, you can copy a DataSource.

- **Data Preview**

  By choosing  *Show Data Preview* in the editor toolbar or *Window* > *Show View* > *Others* , you can call the *Data Preview for DataSource* view. In the *Data Preview for DataSource* view, the system displays a preview of the active version of the DataSource. Here you can specify how many data records to display in the preview. For hierarchy DataSources, you can select which segment to the data is displayed for.
- **Mass maintenance of field properties**
  If you select multiple fields in the field list, you can use mass maintenance to change the properties of the selected fields. If the selected fields have different values for a particular property, this is indicated by the expression <multiple values> or the corresponding flagging of a checkbox. A change made to a property applies to all selected fields. If a property cannot be changed for one of the selected fields, the property cannot be changed in mass maintenance. If the selected fields have different data types, no information is displayed for the length, the number of digits, and the number of decimal places.
- Restricted selection of the delta field (depending on the type of the generic delta)
  On the *Extraction* tab, under *Delta Properties*, the only fields now available for the selection of the delta field for the generic delta are those that are valid for the selected type of the delta field (*Generic Delta*). For dates, only fields with the data type DATS are displayed, while for time stamps only fields of data type DEC or NUMC are displayed. For numeric pointers, all fields are displayed.
- **Property lowercase has been renamed as Case-Sensitive**
  The field property *case-sensitive* can be edited for the following DataSource data types: CHAR, SSTR, STRG, LANG, CUKY und UNIT. The system checks the data type of the DataSource field, not the data type of the source field. If the data type of a type that does not support the property is changed into a type that does support the property, the *Case-Sensitive* property is selected by default. If the data type is changed in such a way that it does not support the *Case-Sensitive* property, the selection is removed.
- Copy technical name in the properties view
  The *Properties* view displays a number of links. You can copy the linke texts via the context menu for the link.

Related Information

Creating DataSources [page 471]

6.5.9 Open Hub Destination (Enhanced)

When you create an open hub destination with type *File*, you can choose *HDFS File* as the destination to store the file in the Hadoop server with.

Related Information

Selecting Files as Destinations [page 453]

6.5.10 Enhanced Master Data Update (changed)

For characteristics with attributes and/or texts, you can use enhanced master data update. This enables parallel loading of data. The characteristic is given an inbound table and a change log as additional tables.

Related Information

Master Data Update [page 58]

6.5.11 Authorizations (Changed)

The following changes apply for authorizations:

- New authorization object in object class RS:
  S_RS_DOCA: Authorizations for working with the BW Accelerator Monitor.
- Changes authorization objects in object class RS:
  S_RS_IOBJA: Authorizations for working with individual InfoObjects with InfoAreas and their subobjects. This authorization object has been enhanced to include processing and activation of master data. It thus replaces authorization object S_RS_IOMAD.
- Obsolete authorization objects in object class RS:
  S_RS_IOMAD Authorizations for processing master data in the Data Warehousing Workbench
- The description of object class RS has changed.
- The user roles delivered by SAP have been modified to take account of the changed tasks in SAP BW/4HANA.

6.5.12 Version Comparison (New)

You can compare two versions with one another. You can compare two versions of an object in the same system or a version of an object in one system and the active version of the same object in a different system.

The version comparison is currently supported for CompositeProviders, DataStore objects (advanced) and InfoObjects.

Related Information

Comparing Object Versions [page 34]

6.6 Version 1.18

This section describes all new functions, enhancements and changes in connection with BW Modeling Tools. New back-end-specific functions are available in back end version SAP BW/4HANA 1.0 SPS 04.

6.6.1 Query: Assignment to Role (New)

If you want to make sure that users with particular role authorizations can see a particular query, you can assign this query to the corresponding roles.

Related Information

Publishing to Roles [page 374]

6.6.2 Query: Linked Component (New)

A linked component can be automatically synchronized whenever changes are made to the corresponding source component.

Example scenario: You have two highly similar InfoProviders, IP_A and IP_B. You have created the query Q_A for IP_A. You now want to create the query Q_B for InfoProvider IP_B. You want this query to be very similar to query Q_A and to be automatically adjusted whenever changes are made to query Q_A.

To do this, you use the link component concept: You create the linked target query Q_B for source query Q_A. This is more than just a copy, as the system also retains the mapping information. This mapping information makes it possible to synchronize the queries.

Related Information

Linked Component [page 140]
Creating a Link Component [page 445]
Apply Changes from the Source Component [page 446]
Changing a Target Component [page 447]

6.6.3 InfoProvider Views in the BW Modeling Tools (Enhanced)

New functions have been added to the *InfoProvider* view.

- Where-Used List
  In the context menu of an InfoObject or a reusable query component (such as a calculated key figure) in the *InfoProvider* view, you can also call the where-used list for the object.
- To hierarchically group calculated and restricted key figures of an InfoProvider, you previously had to create a directory structure in the maintenance screen for the hierarchical display of the reusable key figures (transaction RSZTREE). This function is now available in the *InfoProvider* view.

Related Information

InfoProvider View [page 16]

6.6.4 Automatic Object Check During Processing in the BW Modeling Tools (New)

In the settings for the BW Modeling Tools, you can define which objects should be automatically checked during processing in the editor. The error messages for the objects selected during processing are displayed directly in the *Problems* view.

Related Information

Settings for BW Modeling Tools [page 37]

6.6.5 Wizard for Creating DataSources (Changed)

The wizard for creating DataSources has been revised and given a more user-friendly design. You can now also use the wizard to create file DataSources.

Related Information

Creating a DataSource Using the Wizard [page 472]

6.6.6 Wizard for Creating Source Systems (Changed)

During creation of SAP HANA source systems with connection type SAP HANA Smart Data Access in the *Select Path* Wizard step, folder hierarchies are now displayed in the result list for sources with a hierarchical folder structure. You can select the path prefix from these folder hierarchies.

Related Information

Creating a Source System of Type SAP HANA Smart Data Access [page 461]

6.7 Version 1.17

This section describes all new functions, enhancements and changes in connection with BW Modeling Tools. New back-end-specific functions are available in back end version SAP BW/4HANA 1.0 SPS 02.

Related Information

Creating Source Systems in BW Modeling Tools (new) [page 575]

6.7.1 Creating Source Systems in BW Modeling Tools (new)

It is now possible to create source systems in BW Modeling Tools by using a wizard and editor.

Related Information

Creating a Source System [page 455]

6.8 Version 1.16

This section describes all new functions, enhancements and changes in connection with BW Modeling Tools. The back-end-specific functions are available in back end version SAP BW/4HANA 1.0 SPS 01.

Related Information

Navigation Attributes in CompositeProviders (Changed) [page 576]
Big Data Source System (New) [page 576]
Data Transfer Process and Transformations in Project Explorer View (New) [page 577]
Creating a Query: Defining Priorities (New) [page 577]

6.8.1 Navigation Attributes in CompositeProviders (Changed)

In CompositeProviders you can now activate navigation attributes, if you have set the *Usage of System-Wide Unique Name* flag under *Association*. In this case, new names are generated for the navigation attributes.

Related Information

Activating Navigation Attributes [page 306]

6.8.2 Big Data Source System (New)

The Big Data source system provides central and unified access to Big Data sources. Access is possible via SAP HANA Smart Data Access and the SPARK SQL (DESTINATION) adapter.

Related Information

Transferring Data with a Big Data Source System [page 161]

6.8.3 Data Transfer Process and Transformations in Project Explorer View (New)

The integration of data transfer processes and transformations into BW modeling tools has been improved.

Data transfer processes and transformations are now displayed in the *BW Repository* node of the *Project Explorer* view under the relevant target. In *BW Repository* node, you can create a data transfer process or a transformation via the context menu on the target object. In addition, you can call various functions for data transfer processes and transformations in the *BW Repository* and then copy or delete data transfer processes and transformations.

6.8.4 Creating a Query: Defining Priorities (New)

If you use two structures in a query, the system takes into account the properties, which have been specified (for the structures or for specific cells) as cross products of two structure elements. The system takes the properties into account according to the order in which the properties are specified. The last specification made has priority. On the *Priorities* tab, you can modify the priority handling by explicitly specifying which property is actually taken into account. This is a power user function.

Related Information

Defining Priorities [page 360]
Tab: Priorities [page 361]

6.9 Version 1.15

This section describes all new functions, enhancements and changes in connection with BW Modeling tools.

Related Information

Data Flow (new) [page 577]

6.9.1 Data Flow (new)

Data flows for SAP BW/4HANA and their objects (InfoProviders for example) and relationships between each another (transformations for example) can be displayed, created, edited and documented in a graphical user

interface in the BW Modeling tools. The objects and the relationships between them are displayed as noded with connections in a grid. A data flow can be created as a data flow object (object type of the the TLOGO framework in SAP BW/4HANA).

The data flow is the central and clear entry point for modeling in SAP BW/4HANA. You can use the data flow to model objects and their relationships with each other for a particular scenario.

You can create a data flow object in a graphical editor and benefit from its advantages. For example, you can start by creating a logical data flow (blueprint) and then create BW objects, transformations and loading processes later on, in order to persist data in the metadata tables. You do not necessarily require a data flow object to model objects and relationships in data flows in a graphical editor.

Related Information

Data Flow [page 41]
Working with Data Flows [page 197]

7 More Information

The following sections provide information about important server-side functions for query definition.

Related Information

Rules for Standard Aggregation and Exception Aggregation [page 579]
Formula Calculation [page 583]

7.1 Rules for Standard Aggregation and Exception Aggregation

The following table provides an overview of the rules for standard aggregation and exception aggregations in the BW system.

Use

Abbreviation	Aggregation Rule	Description
AV0	Average (Values not Equal to Zero) (*Average of All Values < > 0*)	If you aggregated with the reference characteristic, the average of the column values is displayed as not equal to zero in the result line.
AV1	Average (Weighted with Number of Days) (*Average Weighted with Calendar Days*)	If you aggregate with the reference characteristic, the average of the column values weighted with the number of days is displayed in the result line.
AV2	Average (Weighted with Number of Work Days; Factory Calendar) (*Average Weighted with Working Days*)	If you aggregate with the reference characteristic, the average of the column values weighted with the number of working days is displayed in the result line.

Abbreviation	Aggregation Rule	Description
AVG	Average (All Values) (*Average of All Values*)	The average of all values is displayed.
CN0	Counter (Values not Equal to Zero) (*Count All Values <> 0*)	The number of values <> zero is displayed in the result line.
CNT	Counter (All Values) (*Count All Values*)	The number of available values is displayed in the result line.
FIR	First Value (*First Value*)	The first value in relation to the reference characteristic is displayed in the result line.
LAST	Last Value (*Last Value*)	The last value in relation to the reference characteristic is displayed in the result line.
MAX	Maximum (*Maximum*)	The highest value of all displayed values in this column is shown in the result line.

Abbreviation	Aggregation Rule	Description
MED	Median (*Median*)	The middle value (central value) of a set sorted in ascending order is displayed: • For a set with an uneven number of elements, the result is the middle value. **Note** For a set of five elements, the result is the value of the third element. Suppose you have the following values: 1, 12, 2, 4, 24. When the values are sorted in ascending order (1, 2, 4, 12, 24), the median is the third middle value, which is 4. • For a set with an even number of elements, the result is the average of the two middle values. **Note** For a set of six elements, the result is the average of the third and fourth elements. Suppose you have the following values: 1, 12, 2, 4, 24, 6. When the values are sorted in ascending order (1, 2, 4, 12, 24, 6), the median is the average of the values 4 and 12, which is 8.
MIN	Minimum (*Minimum*)	The lowest value of all displayed values in this column is shown in the result line.
NGA	(*No Aggregation of Posted Nodes Along the Hierarchy*)	No aggregation of postable nodes along the hierarchy.
NHA	(*No Aggregation Along the Hierarchy*)	No aggregation along the hierarchy.
NO1	(*Exception If More Than One Record Occurs*)	The result of the aggregation is the value of the record if exactly one record goes into the aggregation. Otherwise, the result is the special value NOP.

Abbreviation	Aggregation Rule	Description
NO2	(*Exception If More Than One Value Occurs*)	If only records with the same value go into the aggregation, the result of the aggregation is this value. If the records have different values, then the result is the special value NOP.
NOP	(*Exception If More Than One Value <> 0 Occurs*)	Behaves like NO2 with the difference that records with the value 0 are ignored. If all records have the value 0, the result is 0.
SLC	Simple linear regression: (empirical) correlation coefficient *Simple Linear Regression: Correlation Coefficient*)	The simple linear regression is a static method, whereby a straight line is drawn through a set of n points so that the vertical distance between the points and the straight line is as small as possible. With exception aggregation SLS, the empirical correlation coefficient is displayed, which specifies how accurate the estimated regression line is (in other words, how far the points in the point set are from the straight line). The value of the correlation coefficient is between -1 and +1.
SLI	Simple linear regression: Y Point of Intersection *Simple Linear Regression: Y-Intercept*)	The simple linear regression is a static method, whereby a straight line is drawn through a set of n points so that the vertical distance between the points and the straight line is as small as possible. With exception aggregation SLI, the value of the point of intersection between the regression line and the y axis (for x=0) is displayed.
SLS	Simple linear regression: Gradient *Simple Linear Regression: Slope*)	The simple linear regression is a static method, whereby a straight line is drawn through a set of n points so that the vertical distance between the points and the straight line is as small as possible. With exception aggregation SLS, the gradient of the regression line is displayed.

Abbreviation	Aggregation Rule	Description
STD	Standard Deviation (*Standard Deviation*)	After aggregating with the reference characteristic, the standard deviation of the displayed values is displayed in the results row.
SUM	Summation (*Total*)	The total of all values in this column is shown in the result line.
VAR	Variance (*Variance*)	After aggregating with the reference characteristic, the variance of the displayed values is displayed in the results row.

7.2 Formula Calculation

Use

There are always two operands connected with an operator in a formula.

> ⓘ Note
>
> On the frontend formulas are often also called key figures.

The following objects can be operands in a formula:

- Basis Key Figures
- Calculated Key Figures
- Restricted Key Figures

There are the following types of operators in the BW system:

- Basic Functions
- Percentage Functions
- Data Function
- Mathematical Functions
- Trigonometrical Functions
- Boolean Operators

More Information

- Basic Functions [page 584]

- Percentage Functions [page 584]
- Data Function [page 587]
- Mathematical Functions [page 598]
- Trigonometrical Functions [page 600]
- Boolean Operators [page 601]

7.2.1 Basic Functions

Use

Some important operators from the area of basic functions are listed in the following table:

Operator	Operator Name	Description
+	Plus	Adds Both Operands
-	Minus	Subtracts Second Operand from First
*	Multiply	Multiplies Both Operands
/, DIV	Divide	Divides First Operand by Second
**	High	Square an operand with another, for example, 4**2 means 4 high 2 results in 16.

7.2.2 Percentage Functions

Use

You can use the following percentage functions when defining a formula or a calculated key figure as an operator.

Features

Operator	Operator Name	Description
%	Percentage Deviation	`<Operand1> % <Operand2>` Specifies the percentage deviation of Operand1 to Operand2. Identical to the formula `100 * (<Operand1> - <Operand2>) / abs(<Operand2>)` if <Operand2> is not equal to 0, as well as with the error value for `'Division by 0'` if <Operand2> equals 0.
%A	Percentage	`<Operand1> %A <Operand2>` Specifies the percentage amount of Operand1 to Operand2. Identical to the formula `100 * (<Operand1> - <Operand2>) / abs(<Operand2>)` if <Operand2> is not equal to 0, as well as with the error value for `'Division by 0'` if <Operand2> equals 0.
%_A	Percentage Share with Signed Base Value (%_A)	`<Operand1> %_A <Operand2>`
%CT	Percentage of Total	`%CT <Operand>` Specifies how high the percentage amount is with regard to the result. Result here means the result of the aggregation at the next-highest level (intermediate result).
%GT	Percentage of Total of the Grand Total	`%GT <Operand>` Specifies how high the percentage amount is with regard to the grand total. Grand total here means the result of the aggregation at the highest level of the list. The dynamic filters (filters that were not already defined in the Query Designer) are also included in the calculation of the grand total.

Operator	Operator Name	Description
%RT	Percentage of Total of Query Total	`%RT <Operand>` The same as the percentage function %GT; In contrast to the calculation of the grand total, the dynamic filters are **not** included in the calculation of the query result, that is, it is always normalized to the same size, irrespective of the dynamic filter and navigation status.
%XT	Percentage Share of Result Along the Columns	`%XT <Operand>`
%YT	Percentage Share of Result Along the Rows	`%YT <Operand>`

Example

Percentage Deviation (%)

`'Planned Revenue' % 'Actual Revenue'` specifies the percentage by which the actual revenue exceeds the planned revenue.

Percentage Amount (%A)

`'Fixed Costs' %A 'Costs'` specifies how high the fixed cost part is of a product total cost.

Percentage of Total of Result (%CT)

Year	Region	Revenue	%CT 'Sales'
1999	North	30	50
	South	30	50
	Result 1999	60	33.3333
2000	North	60	50
	South	60	50
	Result 2000	120	66.6667
Overall Result		180	100

Percentage of Total of the Grand Total (%GT)

Year	Region	Revenue	%GT 'Sales'
1999	North	30	16.6667
	South	30	16.6667
	Result 1999	60	33.3333
2000	North	60	33.3333
	South	60	33.3333
	Result 2000	120	66.6667
Overall Result		180	100

Percentage of Total of Query Total (%RT)

Year	Region	Revenue	%RT 'Sales'
1999	North	30	16.6667
	South	30	16.6667
	Result 1999	60	33.3333
Overall Result		60	66.6667

The year 1999 is in the dynamic filter. The 100% refers to the revenue 180 that cannot be seen in the table (including Jahr 2000) (see %GT).

7.2.3 Data Functions

Use

When defining a formula or calculated key figure, you can use the following data functions as operators.

Features

Operator	Operator Name	Description
COUNT	Counter	`COUNT(<Expression>)` Returns value 1 if the expression <> identified by <Expression> results in 0, otherwise 0.
CMR	Current Member Result	`CMR(<Expression>)`
DATE	Casting Operator Date	`DATE` The value is given as the number of days starting from 01.01.0000 and then displayed as a date. The value can be between 0 and a million. This which allows for a date range between 01.01.0000 and 27.11.2738.
DELTA	Delta	`DELTA(<Expression>)` Returns value 1 if the expression identified by <Expression> results in 0, otherwise 0. **⚠ Caution** This function is identical to the NOT operator and should not be used anymore.
FIX	FIX	Fixes the context of the aggregation by preventing the exception aggregation from using the GROUP-BY clause. The constant selection is therefore made before the formula calculation (see Fix Operator [page 592]).
NDIV0	No DIV0	`NDIV(<Expression>)` Results in 0 if the expression identified by <Expression> is caused by a division by 0 during calculation. Otherwise, the result is the value of the expression. This is used in order to avoid error messages or so that you can continue calculating with a defined result.

Operator	Operator Name	Description
NODIM	Value without Units or Currencies	`NODIM(<Expression>)` Returns purely numerical values of <expression> and suppresses units and currencies.
NOERR	No Error	`NOERR(<Expression>)` Results in 0 if the calculation of <expression> causes an arithmetical error. Otherwise, the result is the value of the expression. This is used in order to avoid error messages or so that you can continue calculating with a defined result.
SCAL	Constant as Scalar	Constant as scalar size, meaning that it is characterized solely by its numerical value; existence based on other operands
SUMCT	Norming Operation Result	`SUMCT <Operand>` Returns the result of the operand at the next-highest level (subtotal) (see also Percentage Functions [page 584], %CT).
SUMGT	Norming Operation Grand Total	`SUMGT <Operand>` Returns the grand total of the operand (see also Percentage Functions [page 584], %GT).
SUMRT	Norming Operation Query Result	`SUMRT <Operand>` Returns the query result of the operand (see also Percentage Functions [page 584], %RT). Unlike the process for calculating the overall result, dynamic filters are not used in the calculation of the query result. This means that the calculation always normalizes to the same value, irrespective of the filter status and the navigational state.
SUMXT	Subtotal Along the Columns	`SUMXT <Operand>`

Operator	Operator Name	Description
SUMYT	Subtotal Along the Rows	`SUMYT <Operand>`
TIME	Casting Operator Time	`TIME` The `TIME` operator turns any key figure or (sub)formula into a key figure of type `TIME`. The system removes the decimal digits, interprets the value as seconds, and returns the value as +-hh:mm:ss. **Example** -4812 for example is therefore displayed as `-1:20:12`. A number is expressed as the number of seconds from 00:00:00 and is expressed in hours, minutes, and seconds. A key figure of type `TIME` with the value 0 is displayed as `00:00:00`. Negative times are allowed: For a negative number the system calculates backwards from 00:00:00. You can enter any number of hours. After 24 hours, the system does not restart again at 0.
VECT	Constant as Vector	Constant as vector; existence like a posted key figure

Example

NOERR (x): Equal to 0 for undefined calculations, otherwise x

```
NOERR(SQRT(-1)) = 0
```

Result (SUMCT)

Year	Region	Revenue	SUMCT 'Revenue'
1999	North	30	60
	South	30	60

Year	Region	Revenue	SUMCT 'Revenue'
	Result 1999	60	180
2000	North	60	120
	South	60	120
	Result 2000	120	180
Overall Result		180	180

Grand Total (SUMGT)

Year	Region	Revenue	SUMGT 'Revenue'
1999	North	30	180
	South	30	180
	Result 1999	60	180
2000	North	60	180
	South	60	180
	Result 2000	120	180
Overall Result		180	180

Query Result (Report Total) (SUMRT)

Year	Region	Revenue	SUMRT "Revenue"
1999	North	30	180
	South	30	180
	Result 1999	60	180
Overall Result		60	180

The year 1999 is in the dynamic filter. The missing revenue 120 comes from the year 2000 (see SUMGT).

Process Value as Time (TIME)

This function is mainly intended for offset calculations. If you want to calculate 4 hours for time key figures for example, you can use the `TIME` operator.

7.2.3.1 FIX Operator

The FIX operator is a data function that fixes the context of the aggregation. It prevents the exception aggregation from passing on its reference characteristics to the GROUP BY clause of the preceding aggregation and calculation steps. This allows execution of the constant selection before calculation of the formula, as expected.

Use

There is a correlation between exception aggregation and constant selection: The data that flows into the exception aggregation is grouped by the reference characteristic of the exception aggregation, for example by characteristic *Product*. In conjunction with a constant selection, this behavior means that the constant selection by *Product* is basically rendered ineffective. This makes sense in scenarios where the exception aggregation and constant selection are performed using **different** characteristics.

> ### Example
>
> - Constant selection: 1% of sales with constant selection by all products,
> - Exception aggregation: Number of years in which the sales per product is greater than 1% of total sales,

This is different is the exception aggregation and constant selection are performed using **the same** characteristic.

> ### Example
>
> - Constant selection: 1% of sales with constant selection by all products,
> - Exception aggregation: Sales by product > percentage of sales with constant selection by all products

The following example serves to illustrate that, in this case, the correct execution of aggregation and constant selection in the analytic manager can cause unexpected results from the customer's perspective. The customer might want to specifically influence the result on which the formula exception aggregation is executed at the end.

A query is used to calculate how many products have sales of over 1% of the total sales for all products.

The query has the following key figures:

- The basic key figure is *Sales*.
- The second key figure is the restricted key figure *Sales with constant selection on characteristic Product*. This means that the key figure shows the sales for all products, not just for the product in the filter (Product A).
- The third key figure is calculated using the formula *1% of sales with constant selection by Product*. The key figure *Sales with constant selection on characteristic Product* is multiplied by 0.01 here.
- The fourth key figure is also calculated using a formula: This is a binary comparative operation between *Sales* and *1% of sales with constant selection by Product*. The system should return 1 if *Sales* is greater than *1% of sales with constant selection by Product*. Otherwise it should return 0. This key figure contains the exception aggregation SUM by Product.

The following table shows the query result with an unexpected result from the customer's perspective for key figure *Sales > Sales with constant selection by Product*:

Query Result with Unexpected Result for Key Figure Sales > Sales with Constant Selection by Product

Product	Sales	Sales with Constant Selection by Product	1% of Sales with Constant Selection by Product	Sales > Sales with Constant Selection by Product
A	EUR 450	EUR 100,000	EUR 1000	1
B	EUR 1200	EUR 100,000	EUR 1000	1
Total	**EUR 1650**	**EUR 100,000**	**EUR 1000**	**2**

The analytic manager returns 1 as the result for Product A. As the sales value of 450 is less than als 1000, the expected result would actually be 0. How did this happen?

The analytic manager calculates according to the following formula:

```
SUM( Sales > ( 0.01 * (Sales with constant selection by product) ) by product
```

As the fourth key figure is calculated using a formula, there is a formula exception aggregation here: The exception aggregation SUM is executed on the result of the formula calculation. In accordance with the sequence of aggregation calculations in the analytic manager, formula exception aggregations are executed at the end. This means that reference characteristic *Product* in the GROUP-BY clause is contained in every part of the formula subtree. The exception aggregation therefore overwrites the constant selection here. This is because the following rules applies for the constant selection:

The constant selection only affects the selection and has no effect on the order in which the aggregation and formula calculation are processed. The constant selection is **not** executed before aggregation.

Even if all products were part of the subtotal in accordance with the constant selection, the result can never be correct, as the analytic manager calculates with the sales of the products rather than with the percentage from the constant selection. As X > 0,01 * X, the result is always positive.

- For Product A: 450 > 450*0,01. In this case, the result value "1" is not as expected, as the individual product sales value is smaller than the percentage sales with constant selection over all products.
- For Product B: 1200 > 1200*0.01. In this case, the result value "1" is only as expected because as the individual product sales value is greater than the percentage sales with constant selection over all products.

To prevent the constant selection from being ineffective in scenarios like this, you should use the FIX operator. The FIX operator has the task of excluding the GROUP-BY clause during calculation of the exception aggregation. This allows execution of the constant selection before calculation of the formula. This ensures that the expected result is achieved.

Query Result with FIX Operator

Product	Sales	Sales with Constant Selection by Product	1% of Sales with Constant Selection by Product	Sales > Sales with Constant Selection by Product
A	EUR 450	EUR 100,000	EUR 1000	0
B	EUR 1200	EUR 100,000	EUR 1000	1
Total	**EUR 1650**	**EUR 100,000**	**EUR 1000**	**1**

If the FIX operator is used in the example above, the analytic manager calculates according to the following formula:

```
SUM ( Sales > FIX ( 0.01 * (Sales with constant selection by product) ) by product
```

The system works as follows:

- The system checks which exception aggregation rule is next in the sequence and asks for the reference characteristic.
- It aggregates by the characteristic in accordance with the aggregation rule.
- This single value then serves as the basis for the following calculations:

The query result now provides the expected values.

> ⚠ Caution
>
> Note that the FIX operator only influences one individual exception aggregation. It does this by excluding the GROUP-BY information for the reference characteristic used in the formula. With nested formulas, other formulas are not affected.

Related Information

FIX Operator and Various Aggregations [page 594]
FIX Operators and Various Constant Selections and Aggregations [page 596]
FIX Operator and Restricted Key Figures [page 596]
FIX Operator and Restricted Constants [page 598]

7.2.3.1.1 FIX Operator and Various Aggregations

The following examples show how the system performs a calculation if the FIX operator is in an exception aggregation or if an exception aggregation is in the FIX operator.

The FIX operator only affects the reference characteristic used in the formula of a particular exception aggregation. In the example below, we expand on the example for using the FIX operator with reference characteristic *Calendar Month*. The following key figure is added: *SUM (Sales > 1% of sales with constanct selection by product) by calendar month*. This means that we add an additional exception aggregation, SUM by calendar month. To make the table clearer, we restrict the calendar month to the months 04.2004 and 05.2004.

FIX Operator and Various Aggregations, Table 1

Calendar Month	Product	Sales	Sales with Constant Selection by Product	1% of Sales with Constant Selection by Product	Sales > 1% of Sales with Constant Selection by Product	Sales > 1% of Sales with Constant Selection by Product by Calendar Month
04.2004	A	EUR 40	EUR 8,000	EUR 100	0	0
	B	EUR 120	EUR 8,000	EUR 80	1	1
	Total	EUR 160	EUR 8,000	EUR 80	1	1

Calendar Month	Product	Sales	Sales with Constant Selection by Product	1% of Sales with Constant Selection by Product	Sales > 1% of Sales with Constant Selection by Product	Sales > 1% of Sales with Constant Selection by Product by Calendar Month
05.2004	A	20	EUR 10,000	EUR 100	0	0
	B	320	EUR 10,000	EUR 100	1	1
	Total	340	EUR 100,000	EUR 100	1	1
Total		**500**	**EUR 100,000**	**EUR 180**	**1**	**2**

The formula that the analytic manager executes is as follows:

```
SUM( SUM( Sales > FIX( 0.01 * (Sales with constant selection by product) ) by
product ) by calendar month
```

The FIX operator only removes the GROUP-BY clause for *Product*, as this characteristic is the reference characteristic of the subformula. The GROUP-BY clause is retained for the *Calendar Month* characteristic. This explains the deviating result in the total.

What would happen if an exception aggregation were defined in the FIX operator?

In the following example, the reference key figure was changed: It is now 1% of the average sales of all products per calendar month. The formula that the analytic manager executes is as follows:

```
AVG( 0,01* (Sales with constant selection by product)) by calendar month
```

The check is performed as in the previous example, but with the new reference characteristic:

```
SUM( Sales > FIX( AVG( 0,01 * (Sales with constant selection by product)) by
calendar month) ) by product
```

FIX Operator and Various Aggregations, Table 2

Product	Sales	Sales with Constant Selection by Product	1% of Sales with Constant Selection by Product	Sales > 1% of Sales with Constant Selection by Product
A	EUR 60	EUR 18,000	EUR 93	0
B	EUR 440	EUR 18,000	EUR 93	1
Total	**EUR 500**	**EUR 18,000**	**EUR 193**	**1**

This example illustrates that the FIX operator prevents the inner aggregation from being executed as defined by the inner formula. It does not affect the aggregation of the formula result.

7.2.3.1.2 FIX Operators and Various Constant Selections and Aggregations

The following examples show how the system performs a calculation if the FIX operator is in an exception aggregation or if an exception aggregation is in the FIX operator.

We take the example of using the FIX operator here and add the calculated reference key figure *Sales with Constant Selection by Product, Calendar Month*. The restriction to calendar month in the previous example is thus omitted. The comparison figure, Sales > 1‰, is now given a nested exception aggregation again:

```
SUM ( SUM (X) by Product ) by Calendar Month
```

The following values are calculated in the result:

FIX Operators and Various Constant Selections and Aggregations

Calendar Month	Product	Sales	Sales with Constant Selection by Product	1‰ of Sales with Constant Selection by Product	Sales > 1‰ of Sales with Constant Selection by Product, Calendar Month
04.2004	A	EUR 40	EUR 100,000	EUR 100	0
	B	EUR 120	EUR 100,000	EUR 100	1
	Total	EUR 160	EUR 100,000	EUR 100	1
05.2004	A	20	EUR 100,000	EUR 100	0
	B	320	EUR 100,000	EUR 100	1
	Total	340	EUR 100,000	EUR 100	1
Total		**500**	**EUR 100,000**	**EUR 100**	**2**

The formula that the analytic manager uses for the column on the far right is as follows:

```
SUM( FIX( SUM( Sales > FIX( 0.001 * (Sales with Constant Selection by Product,
Calendar Month) ) by Product ) ) Calendar Month
```

Every FIX operator removes a reference characteristic: The outer one removes the GROUP-BY clause for Calendar Month, the inner one removes the GROUP-BY clause for Product. If there are multidimensional aggregations with nested formulas, the formula must be defined as in this example. Having two FIX operators ensures that both reference characteristics are removed. Note however that the outer FIX operator also removes the GROUP-BY clause for Calendar Month, which is also needed for the calculation of the sales figure in the inner formula. In cases like this, the only way to obtain the right result is to always calculate against the full reference key figure and with the basic key figure in its full granularity.

7.2.3.1.3 FIX Operator and Restricted Key Figures

The following is an example without constant selection and serves to show that the FIX operator can also be used in cases like this.

Working without constant selection offers no benefits however, as you obtain the same results as you would have without the FIX operator.

The following example has the following key figures:

- The basic key figure is *Sales* again.
- The second key figure is the restricted key figure *Sales (Product A), restricted to product "A"*.
- The third key figure is the restricted key figure *Sales (Product A, Product B), restricted to the products "A" and "B"*.
- The fourth key figure is calculated using a formula: *Sales (Product A) / Sales (Product A, Product B) with Exception Aggregation SUM by Product*.

FIX Operator and Restricted Key Figures, Table 1

Product	Sales	Sales (Product A)	Sales (Product A, Product B)	Sales (Product A) / Sales (Product A, Product B) by Product
A	EUR 450	EUR 450	EUR 450	1
B	EUR 1200		EUR 1200	
Total	**EUR 1650**	**EUR 450**	**EUR 1650**	**1**

The formula that the analytic manager executes is as follows:

```
SUM( Sales (Product A)/ Sales (Product A, Product B) ) by Product
```

Incorporating the FIX operator, this formula is:

```
SUM( Sales (Product A)/FIX( Sales (Product A, Product B) ) ) by Product
```

FIX Operator and Restricted Key Figures, Table 2

Product	Sales	Sales (Product A)	Sales (Product A, Product B)	SUM(Sales (Product A)/FIX(Sales (Product A, Product B))) by Product
A	EUR 450	EUR 450	EUR 450	1
B	EUR 1200		EUR 1200	
Total	**EUR 1650**	**EUR 450**	**EUR 1650**	**0,27**

The FIX operator causes *Sales (Product A) / Sales (Product A, Product B)* to be aggregated with the standard aggregation, as there is no exception aggregation defined in the inner formula. As the FIX operator removes the GROUP-BY clause for Product, the entire formula exception aggregation by product is omitted. The calculation therefore has the same result as it would if no exception aggregation had been defined in the formula.

7.2.3.1.4 FIX Operator and Restricted Constants

The following example illustrates another way of using the FIX operator: Restricted constants

FIX Operator and Restricted Constants

Product	Sales (Product A, Product B, Product C)	10 (Product A, Product B)	SUM (10(Product A, Product B)) by Product	SUM ((SUM(10 (Product A, Product B)) by Product) / (Sales (Product A, Product B, Product C))) by Product	SUM ((SUM(10 (Product A, Product B)) by Product) / FIX (Sales (Product A, Product B, Product C))) by Product
A	EUR 450	10	10	0,022	0,022
B	EUR 1200	10	10	0,008	0,008
C	EUR 1210	10			
Total	**EUR 2860**	**10**	**20**	**0,030**	**0,007**

This example serves primarily to illustate the difference between a formula without exception aggregation and a formula with exception aggregation.

- In a formula without exception aggregation, a constant is always treated as a value.
- In a formula with exception aggregation, the system takes the restriction of a restricted key figure and aggregates accordingly. In a formula with exception aggregation, the system performs the calculation along the drilldown. In other words, the total is calculated from the subtotals. This gives us the value 0.030 for key figure *SUM ((SUM(10 (Product A, Product B)) by Product) / (Sales (Product A, Product B, Product C))) by Product*.

If we want to have a formula with exception aggregation, but the calculation direction of a formula without exception aggregation, i.e. the total of the operands should be be the total of the formula result, we can use the FIX operator: Formula *SUM ((SUM(10 (Product A, Product B)) by Product) / FIX (Sales (Product A, Product B, Product C))) by Product* results in 0.007, calculated from 10 (Product A) / 2860 + 10 (Product B) / 2860 + NULL (Product C) / 2860 = 20/2860. This is the same as the calculation of a formula without exception aggregation, but using a formula with exception aggregation in order to have the restricted key figure as the starting point.

7.2.4 Mathematical Functions

Use

Some important operators from the area of mathematical functions are listed in the following table:

Operator	Operator Name	Description
ABS	Absolute Value	

Operator	Operator Name	Description
CEIL	Round-Up Function	Rounds up to the smallest integer value that is greater than or equal to the operand
DIV	Integer Division	
EXP	Exponential Function	
FLOOR	Round-Down Function	Rounds down to the largest integer value that is smaller than or equal to the operand
FRAC	Decimal Digits of the Operand	Maps the numerical value of the operand to its decimal digits
LOG	Logarithm	
LOG10	Logarithm with Base 10	
MAX	Maximum	Returns the larger of two operands
MAX0	Maximum of 0 and Operand	
MIN	Minimum	Returns the smaller of two operands
MIN0	Minimum of 0 and Operand	
MOD	Modulo	Returns the remainder after the division of two operands
SIGN	Sign of the Operand	
SQRT	Square Root	
TRUNC	Truncate	Integer value of the operand with its decimal digits removed

7.2.5 Trigonometric functions

Trigonometical functions describe the calculation relationships between angles and measurements.

Use

The following functions are supported:

Trigonometric functions

ACOS	Arc Cosine	`ACOS(<Expression>)` Returns the arc cosine of the expression <Expression>.
ASIN	Arc Sine	`ASIN(<Expression>)` Returns the arc sine of the expression <Expression>.
ATAN	Arc Tangent	`ATAN(<Expression>)` Returns the arc tangent of the expression <Expression>.
COS	Cosine	`COS(<Expression>)` Returns the cosine of the expression <Expression>.
COSH	Hyperbolic Cosine	`COSH(<Expression>)` Returns the hyperbolic cosine of the expression <Expression>.
SIN	Sine	`SIN(<Expression>)` Returns the sine of the expression <Expression>.
SINH	Hyperbolic Sine	`SINH(<Expression>)` Returns the hyperbolic sine of the expression <Expression>.
TAN	Tangent	`TAN(<Expression>)` Returns the tangent of the expression <Expression>.

TANH	Hyperbolic Tangent	`TANH(<Expression>)` Returns the hyperbolic tangent of the expression <Expression>.

7.2.6 Boolean Operators

Boolean operators are functions that you can use as operators for defining a formula or a calculated key figure.

Use

The following functions are supported:

Features

Operator	Operator Name	Description
<, <=, <>, ==, >, >=	Relational Operators	`<Expression1> <Operator> <Expression2>` The result is 1 if the expression described by `<Expression1>` in the relationship described by `<Operator>` for `<Expression2>` (i.e. `<Expression1> <Operator> <Expression2>` is true). Otherwise it is 0. Only the number values of `<Expression1> und <Expression2>` are compared without considering the unit. **Example** **Costs < Revenue** returns 1 if the revenue is greater than the costs and 0 if the costs are greater or equal to the revenue.

Operator	Operator Name	Description
AND	Logical AND	`<Expression1> AND <Expression2>` The result is 1 if `<Expression1>` and `<Expression2>` are (both) greater than or less than 0. Otherwise it is 0. Only the number values of `<Expression1>` and `<Expression2>` are compared without considering the unit.
OR	Logical OR	`<Expression1> OR <Expression2>` The result is 1 if `<Expression1> oder <Expression2>` is greater than or less than 0. Otherwise it is 0. Only the number values of `<Expression1> und <Expression2>` are compared without considering the unit.
XOR	Logical Exclusive OR	`<Expression1> XOR <Expression2>` The result is 1 if either `<Expression1> oder <Expression2>` (but not both of them) is greater than or lower than 0. Otherwise it is 0. Only the number values of `<Expression1>` and `<Expression2>` are compared without considering the unit.
NOT	Logical NOT	`NOT <Expression>` The result is 1 if `<Expression>` is 0. Otherwise it is 0. Only the number value of `<Expression>` is compared without considering the unit.

Operator	Operator Name	Description
LEAF	Leaf	The result is 0 on the results line, or the real (inner) node of a hierarchy, and 1 on the elementary line or the leaves of a hierarchy. With this operator you can perform different calculations on results lines and elementary lines.

<table>
<tr><th>Operator</th><th>Operator Name</th><th>Description</th></tr>
<tr><td>IF</td><td>Conditional Calculations</td><td>Conditional calculations can be performed by the IF operator with three operands.
<code>IF(< operand_cond >; < operand_true >; < operand_false >)</code>
Example
IF (Actual Costs > Plan Costs; Plan Costs + Plan Deviation; Actual Costs)
Calculation Logic<table><tr><th>Value of <operand_cond></th><th>Result of the IF Operator</th></tr><tr><td>ERROR, DIV0, OVERFLOW, NOP</td><td><operand_cond></td></tr><tr><td>Greater than or less than 0</td><td><operand_true></td></tr><tr><td>0 or NULL</td><td><operand_false></td></tr></table>Note
If the condition is false, and you want the formula to output a special value, you can add one of the following variables to the formula instead of the third operand <operand_false>:
0NOP_VALUE for the special value NOP
0ERR_VALUE for the special value ERROR
Example
IF(CNT Insured >= 200; Key Figure; 0NOP_VALUE)</td></tr>
</table>

Operator	Operator Name	Description
		Please note the following rules when using the IF operator: • Only comparisons between number values are allowed. • If <operand_cond> has the value 1 (= true) and <operand_false> has an error value (NOP, ERROE, DIV0), the error value will **not** be returned as the result of the IF operator. The same occurs if <operand_cond> does not have the value 1 (= false) and <operand_false> has an error value. • On the basis of <operand_cond>, either <operand_true> or <operand_false> is calculated (excluding the formula exception aggregation). • If <operand_cond> is NULL, <operand_false> will be returned as the result of the IF operator. • The currency or unit of measure of <operand_cond> has no effect on the result of the IF operator.

7.3 Using Precalculated Value Sets (Buckets)

Use

A bucket is a precalculated value set that is stored in a database table and can be used as a filter in an analytic query.

The filter is defined in an analytic query as a variable with processing type replacement path (replacement from a result of a pre-query) and executed at runtime. It is important here that the filter is executed on the database before all the data from the bucket has to be loaded to OLAP processing. This makes for significant runtime improvements if large amounts of data are processed in the bucket.

A variable that refers to buckets can only be used as a filter in characteristic restrictions. It cannot be used in default values, structure elements and the like.

To fill a bucket with data, you can either use a reference table or add the data directly.

Procedure

You are in the Bucket Definition screen (transaction RSBUCKET).

To create a bucket, proceed as follows:

1. Select the characteristic that you want to create a bucket for.
2. Specify the name of the bucket.

   > ⓘ Note
   >
   > Input help is provided for existing buckets.

3. You can also enter a description.
4. If you want to use an existing reference table, select *Use Existing Sources*. You can select the reference table under *Existing Table*. The system reads the data from a specified column in a database table (that is known to the ABAP DDIC).
   1. To view the content of a table, press *Display Table*.

      > ⚠ Caution
      >
      > Note that displaying a large table can be time-consuming if a large number of rows is set as standard for it in the Data Browser (transaction SE16). You do have the option however of refining the display and thus minimizing the number of values that are actually displayed.

   2. Specify an ABAP table as a reference.
   3. Choose either key values (by selecting Table contains keys) or SID values (by selecting Table contains SIDs).
   4. In the *Field Name* in Table field, specify the column that you want the characteristic values to be read from.
5. If you do not want to use an existing reference table, do not select *Use Existing Sources*. You can select the characteristic values under *Characteristic Values of the Bucket*. You can select the values manually, copy them from the buffer or upload them from files.
6. To save your settings, press *Save*.

Result

The system has precalculated the required value set, which is now available in the form of a bucket. In an analytic query, you can use this bucket to define a variable with processing type replacement path (replacement from a query).

Related Information

Replacement from Query [page 130]
Example: Using a Variable with Replacement from Query [page 391]

7.4 CURRENT MEMBER Variables

With a CURRENT MEMBER variable, you can define filters for structure elements of a query; these filters have limits that depend on the value of a characteristic in the drilldown.

This allows you to define queries to cover the following business scenarios for example:

1. **Comparison with previous month**: The query shows the value of a key figure per month together with the value of the key figure in the previous month. These values can be used in formulas, for example to calculate the change or percentage change in comparison with the previous month. You can define the value from the previous month in Query Designer by defining a restricted key figure and restricting the month with the CURRENT MEMBER variable with an offset of -1.
2. **YTD (year to date) analyses**:
   - **YTD (year to date) analyses**: The query shows the aggregated value of a key figure per month starting from the value for January through to the month in the row. You can define this key figure in Query Designer by defining a restricted key figure and restricting the month by a range with January of the year in question as the lower limit and the CURRENT MEMBER variable as the upper limit.
   - **Multi-year YTD (year to date) analyses**: As in the YTD analysis in example 2a, the query displays the aggregated value of a key figure per month. In this case however the lower limit is not fixed. The start of the year is fixed instead. This is taken from the value for the month. As long as the report only involves a single year, there is no difference between this and example 2a. As soon as the report involves a second year however, the figures are only cumulated by year.
3. **Rolling window**: The query shows the aggregated value of a key figure over the last three months starting with month in the row. You can define this key figure in Query Designer by defining a restricted key figure and restricting the month by a range with the CURRENT MEMBER variable with offset -2 as the lower limit and the CURRENT MEMBER variable as the upper limit.
4. **Rolling average**: Rather than showing the value aggregated using standard aggregation (as in example 3), this should show the average value over the last three months. Firstly, define a restricted key figure in example 3. Then define a formula equal to this key figure and set exception aggregation "Average“ for the month.

> ⓘ Note
>
> Whereas time characteristic Month is used here, you can take any other time characteristic.

CURRENT MEMBER variables can only be defined for characteristics, which relate to a calendar. These include 0CALDAY, 0CALMONTH, 0FISCPER and other similar characteristics, as well as characteristics that reference these. This function is not possible for other characteristics, especially pseudo time characteristics such as 0CALMONTH2 or 0FISCPER3.

The corresponding variables for the content time characteristics are delivered with the technical content in SAP BW/4HANA. The technidal name of the variables is made up of the prefix 0CURMEM_ and the technical name of the time characteristic, 0CURMEM_CALDAY or 0CURMEM_CALMONTH for example. You can also create a variable of your own.

For a CURRENT MEMBER variable, a *First Value* or *Last Value* operator can be extended in the variable definitions. You also have to select a time characteristics that can be derived from the time characteristic of the variables.

> ### Example
>
> You have defined a characteristic value variable for characteristic 0CALMONTH with replacement path CURRENT MEMBER. You select the *First Value* operator and time characteristic 0CALYEAR. In a result row, a variable of this kind is not replaced by the value itself but by the *First Month of the Year*. This makes it possible to have example 2b.

> ### Note
>
> Note regarding system performance: the system reads the data from the database, in the granularity of the characteristic of the CURRENT MEMBER. In addition, different database access types may be required for various restrictions with CURRENT MEMBER variables.

How it Works

The CURRENT MEMBER variable can be used in restricted key figures as either the lower limit or the upper limit in combination with an offset. However, only one restriction with a CURRENT MEMBER variable - with a maximum of one interval - is allowed per restricted key figure.

In a colum (row) with a key figure restricted in this way, a different filter affects every cell displayed in the query result:

- In cells where the characteristic of the CURRENT MEMBER variable has been clearly defined, the CURRENT MEMBER variable is replaced with this value. As this occurs for both the lower limit and the upper limit, a range selection normally results.
- In cellls where the characteristic of the CURRENT MEMBER variables has **not** been clearly defined (in the result rows for example), the other filters on time characteristics are converted to ranges for the characteristic with the CURRENT MEMBER variable. The CURRENT MEMBER variable in the lower limit is replaced by the upper limit of the range, and the CURRENT MEMBER variable in the upper limit is replaced by the lower limit of the range.

  > ### Note
  >
  > For the cells that filters on the time characteristics **cannot** be converted in ranges of the characteristic with the CURRENT MEMBER for, the system returns the value `none exists`.

- If there are any other restrictions for the characteristic in the structure element with the CURRENT MEMBER variable, these are intersected with the restrictions by the CURRENT MEMBER variable. This way of handling deviates from the rule that different restrictions on a characteristic should be linked with a logical OR and a union set should be built accordingly.
- In structure elements with CURRENT MEMBER variables, the system converts global filters into filters for the CURRENT MEMBER characteristic.

  > ### Note
  >
  > If a query is restricted to the month interval January 2013 to December 2013, then a column, which is restricted to CM-1, would still show the value for December 2012 as the value for January 2013.

> ### Example
>
> The value for every cell in the example below can be reproduced by a query, in which the characteristic with the CURRENT MEMBER variable for month contains exactly the filter entered in the cell. The filter in the

query definition is Year = 2013. The CURRENT MEMBER variable is abbreviated as CM in the following table. CM-1 is the variable with offset -1.

Example query for using CURRENT MEMBER variables; the global filter is included.

Month	CM -1	01.2013 - CM	(CM-2) - CM	(CM-2) - CM [02.2013, 12.2013]
01.2013	12.2012	01.2013-1.2013	11.2012-01.2013	empty set
02.2013	01.2013	01.2013-2.2013	12.2012-02.2013	02.2013-02.2013
...	...	...	...	...
Result	**12.2012-11.2013**	**01.2013-12.2013**	**11.2012-12.2013**	**02.2013-12.2013**

Dependency of Query Display on Characteristic Property *Access Type for Result Value*

If the CURRENT MEMBER variable in an interval is used in a structure element, the cells are only displayed in certain circumstances:

- A cell is displayed if there is a posting for the key figure of the time characteristic.
- A cell is displayed if the option has been specified that all values - including unposted values - are displayed for the time characteristic in the drilldown of the query. In the Query Designer, this is specified using the property *Access Type for Result Values* with the value *Master Data*.

Example

Only the following values are posted in an InfoProvider:

- 01.2013 with the value 10
- 06.2013 with the value 15

In this case, the query displays:

Query with restriction of the cells on posted values

Month	Key figure 01.2013 - CM
01.2013	10
06.2013	15
Result	**25**

If the property *Access Type for Result Values* is selected for the characteristic Month with the value *master data*, the query displays the following image:

Query displaying all cells

Month	Key figure 01.2013 - CM
01.2013	10
02.2013	10

Month	Key figure 01.2013 - CM
...	...
05.2013	10
06.2013	25
...	...
12.2013	25
Result	**25**

CM.FirstQuarter is a CURRENT MEMBER variable for 0CALMONTH with the *First Value* operator and time reference 0CALQUARTER. The query appears as follows:

Query with CM.FirstQuarter

Month	Key figure CM.FirstQuarter - CM
01.2013	10
02.2013	10
03.2013	10
04.2013	0
...	...
05.2013	0
06.2013	15
...	...
Result	**25**

Constraints

- A CURRENT MEMBER variable can only be defined for **time characteristics**.
- This function can only be used if the time characteristics of the selected InfoProvider have the same semantics and are thus dependent on each other.
- Time characteristics that **cannot** be expressed as an interval of the characteristic of the CURRENT MEMBER variables are not permitted in the global filter or as a free characteristic.

  > ⓘ Note
  >
  > If the CURRENT-MEMBER variable relates to 0CALMONTH (for example), then 0CALYEAR and 0CALQUARTER (for example) are permitted as free characteristics but 0CALDAY, 0CALWEEK or 0FISCPER are not permitted.

- A CURRENT MEMBER variable can only be used in selections of a structure element or in a restricted key figure, but NOT in the filter for the query.

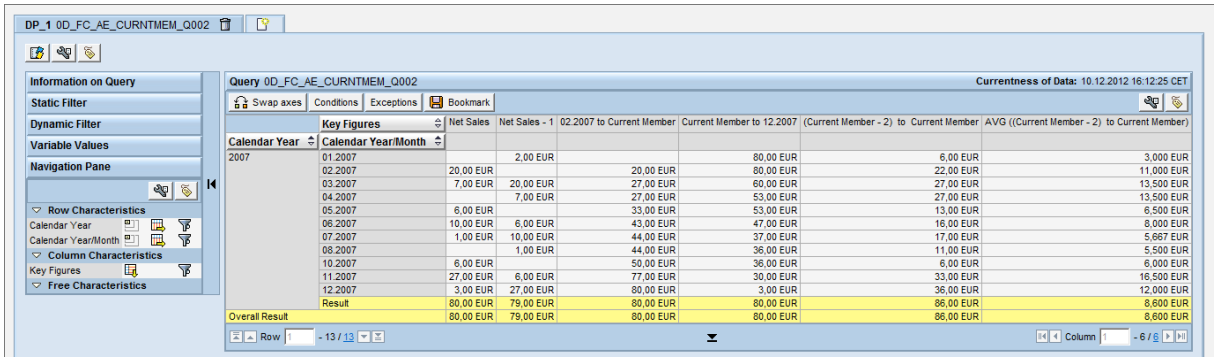
- Each structure element can have just **one** selection with a CURRENT MEMBER variable or with a range made up of CURRENT MEMBER variables. This filter cannot be defined as "excluding“ (only INCLUDING is allowed).
- Due to technical reasons, an upper or a lower limit is needed in the global filter of a query for the characteristic of the CURRENT MEMBER variables, if the variable is used as an upper or a lower limit. If this limit is not set, the system displays a warning and the value `non exists` is shown in the relevant cells.
- Manual planning is **not** possible on structure elements with a CURRENT MEMBER variable.

Related Information

Example Query for Using a CURRENT MEMBER Variable [page 611]
Examples of Using the CMR Operator [page 612]
Defining Variables [page 389]
Replacement from CURRENT MEMBER Variable [page 130]

7.4.1 Example Query for Using a CURRENT MEMBER Variable

The screenshot below shows the query with the technical name 0D_FC_AE_CURNTMEM_Q002:



The query structure contains the following characteristics:

- Calendar Year, restricted to calendar year 2007
- Calendar Year/Month

The query contains the following key figures:

- Net Sales This column shows the Net Sales figure per month.
  Result: Total of all sales figures posted in 2007.
- Net Sales - 1: This column shows the Net Sales figure from the previous month.
  Row 01.2007 contains the sales figure from December 2006.
  Result: The total of all sales posted from December 2006 to November 2007.
- 02.2007 to CURRENT MEMBER (0CALMONTH): This is a year-to-date analysis performed using the CURRENT MEMBER variable 0CURMEM_0CALMONTH. The system calculates the cumulative sales from Februar 2007 up to and including the current month.

Result: Total of all sales figures posted in 2007 from February onwards. As no sales were posted in Januar, this is the total of all sales for the entire year.

- CURRENT MEMBER (0CALMONTH) to 12.2007: The system calculates the sales figure from the current month to December 2007.
  Result: Total of all sales figures posted in 2007.
- (CURRENT MEMBER (0CALMONTH) - 2) to CURRENT MEMBER (0CALMONTH): This is a rolling screen. The system calculates the cumulative sales figure for the last three months.

  > ⓘ Note
  >
  > You can also use this key figure in formulas, for example to calculate the share of the sales figure from a given month in a given quarter.

  Result: Total sales figure from 2006 to December 2007.
- AVG ((CURRENT MEMBER (0CALMONTH) - 2) to CURRENT MEMBER (0CALMONTH)): This key figure is defined using formula exception aggregation AVG for reference characteristic 0CALMONTH. The system calculates the average sales from the last three months.
  Result: Total sales figure from November 2006 to December 2007 dvided by the number of values posted. If sales were only posted in 10 of the 14 months in question, the division is made by 10.

7.4.2 Examples of Using the CMR Operator

If your queries incorporate key fiigures restricted by CURRENT MEMBER variables, you can define formulas using the CMR (CURRENT MEMBER result) operator. This data function does not calculate the results based on the data in the InfoProvider. Instead it calculates the them based on the values moved by the CURRENT MEMBER variable.

The folowing examples will illustrate how the CMR operator works.

Example Query 1

In the following example, the variable with replacement path CURRENT MEMBER refers to the content time characteristic Month (`0CALMONTH`). CMR in the header refers to the CURRENT MEMBER result operator. Sales in December 2006 = 2.00.

Example query 1 with CURRENT MEMBER variables for the month

Calendar Year/Month	Sales	CM-1 - CM	AVG (CM-1 - CM, Month)	AVG (CMR (CM-1 - CM), Month)
01.2007		2,00	2,00	2,00
02.2007	20,00	20,00	20,00	20,00
03.2007	7,00	27,00	13,50	27,00
04.2007		7,00	7,00	7,00
05.2007	6,00	6,00	6,00	6,00

Calendar Year/Month	Sales	CM-1 - CM	AVG (CM-1 - CM, Month)	AVG (CMR (CM-1 - CM), Month)
06.2007	10,00	16,00	8,00	16,00
07.2007	1,00	11,00	5,50	11,00
08.2007		1,00	1,00	1,00
09.2007				
10.2007	6,00	6,00	6,00	6,00
11.2007	27,00	33,00	16,50	33,00
12.2007	3,00	30,00	15,00	30,00
Result	**80,00**	**82,00**	**9,11**	**14,455**

The query contains the following key figures:

- **Sales**: This column shows the Net Sales figure per month. Sales is a key figure in the InfoProvider.
  - Result: Total of all sales figures posted in 2007.
- **(CM-1) - CM**, in other words `(CURRENT MEMBER (0CALMONTH)-1) - CURRENT MEMBER (0CALMONTH)`: This is rolling window. The system calculates the cumulative sales from the last three months.
  - The value for month 07.2007 is 11.00, as `(CM-1) - CM` defines the interval of the months 06.2007 (= CM-1) to 07.2007, and the total for the months of June (10,00) and July (1,00) is exactly 11.00.
  - Result: Total sales figure from 2006 to December 2007.
- **AVG ((CM-1) - CM, Monat)**, in other words `AVG ((CURRENT MEMBER (0CALMONTH) - 1) bis CURRENT MEMBER (0CALMONTH))`: This key figure is defined using formula exception aggregation AVG for reference characteristic `0CALMONTH`. The system calculates the average sales per month based on incoming data from the last two months, i.e. the two-month sales average.
  - The value for month 07.2007 is 5.50, as the system divides the cumulative sales for the months of June to July - 11.00 - by the number of months with incoming data, in this case 2.
  - The value for month 08.2007 is 1.00, as the system divides the cumulative sales for the months of July to August - 1.00 - by the number of months with incoming data, in this case 1.
  - Result: Average sales per month. The system calculates the total sales for all months (82.00) and divides this by the number of months with incoming postings (1 in 2006 and 8 in 2007 - divided by 9).

  > **Note**
  >
  > An example of where average calculations of this kind are used is the 200 day average of sales.

- **AVG (CMR (CM-1 - CM), Monat)**, in other words `CMR ((CURRENT MEMBER (0CALMONTH) - 1) bis CURRENT MEMBER (0CALMONTH))`: This key figure shows the average of the interval values and not the average of the incoming values. This is specified by the CMR operator. This means that the average is not calculate based on the data in the InfoProvider. Instead, it is calculated based on the values moved by the CURRENT MEMBER variable (see `(CM-1) - CM`)). The values from the previous month are included in the range for the current month, meaning that it is not possible to differentiate for individual months between the cumulative values in columns `(CM-1) - CM)` and values changed by the CMR operator in `AVG (CMR (CM-1 - CM), Monat)`. The difference can be seen in the result however.
  - Result: Average sales per interval. The system calculates the total sales from all intervals (159) and divides this by the number of intervals with values (11 in 2007).

Example Query 2

This example shows a query that is used to find out how many months show growth in sales compared to the previous month.

Example Query 2

Calendar Year/Month	Sales for Current Month	Sales for Previous Month	Difference in Sales to Previous Month	Difference in Sales to Previous Month > 0
01.2007		2,00	-2,00	0
02.2007	20,00		20,00	1
03.2007	7,00	20,00	-13,00	0
04.2007		7,00	-7,00	0
05.2007	6,00		6,00	1
06.2007	10,00	6,00	4,00	1
07.2007	1,00	10,00	-9,00	0
08.2007		1,00	-1,00	0
09.2007				0
10.2007	6,00		6,00	1
11.2007	27,00	6,00	21,00	1
12.2007	3,00	27,00	-24,00	0
Result	**80,00**	**79,00**	**1,00**	**5,00**

The query contains the following key figures:

- **Sales for Current Month**: This column shows the Net Sales figure per month. Sales is a key figure in the InfoProvider.
  - Result: Total of all sales figures posted in 2007.
- **Sales for Previous Month**, in other words `Umsatz von (CURRENT MEMBER (0CALMONTH)-1)`: This column shows the Net Sales figure from the previous month.
  - Result: Total sales figure from December 2006 to end of November 2007.
- **Difference in Sales to Previous Month**, in other words `Umsatz (CM) - Umsatz (CM-1)`: The system subtracts the sales for the previous month from the sales for the current month.
  - The value for month 07.2007 is -9,00, as the system subtracts the sales for month 06.2007- 10.00 - from the sales for month 07.2007- 1.00.
  - Result: Difference between results: 80.00 - 79.00 = 1.00.
- **Differenz des Umsatzes zum Vormonat > 0**, in other words `CMR (Umsatz (CM) - Umsatz (CM-1))`: This key figure is defined using the CMR operator. If the condition is met, the system returns 1. Otherwise it returns 0.
  - Result: Total of all positive values, in this case 5.00. 5 months displayed growth in sales when compared to the previous month.

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.
About the icons:

- Links with the icon : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
  - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
  - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon : You are leaving the documentation for that particular SAP product or service and are entering an SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.
The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.

www.sap.com/contactsap

© 2024 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company. The information contained herein may be changed without prior notice.

Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

Please see https://www.sap.com/about/legal/trademark.html for additional trademark information and notices.

THE BEST RUN SAP