Working with the DataStore Object (advanced)

Objectives

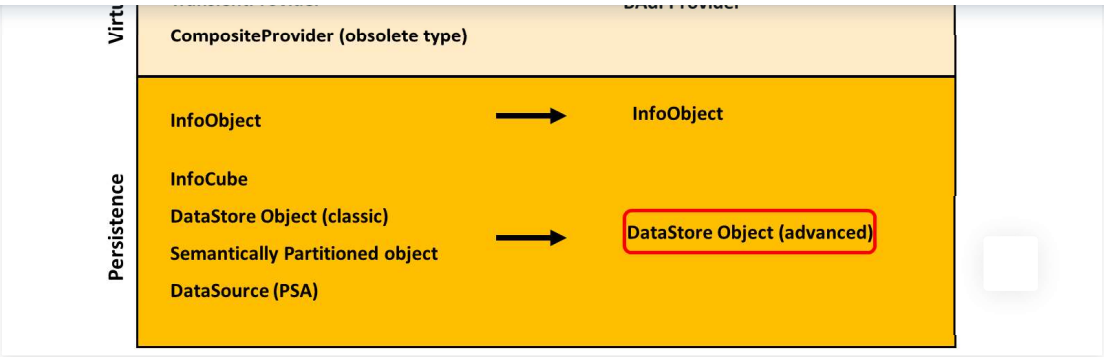
After completing this lesson, you will be able to:

- Create a DataStore Object (advanced)
- Describe how a DataStore Object (advanced) supports common modeling scenarios

DataStore Object (advanced)

Introduction

In SAP BW/4HANA, a DataStore Object (advanced) is the only object for managing persistent transactional data. It is the successor to the DataStore Object (classic) of SAP BW.



As well as replacing the original DataStore Object (classic), the new DataStore Object (advanced) also replaces the InfoCube, Semantically Partitioned Object, and the PSA.

Developers can leverage the DataStore Object (advanced) to model the transactional persistence across multiple layers from data acquisition to data marts. The DataStore Object (advanced) combines the functions of the InfoCube and the DataStore Object (classic) with many extra features, such as modeling with fields and data temperature management.

The DataStore Object (advanced) is more flexible than its predecessor and can contain both fields and InfoObjects in all combinations (fields only, InfoObjects only, or a mixture of both). Any DataStore Object (advanced) can now include up to 1000 fields (key fields/InfoObjects + data fields/InfoObjects) and can include up to 120 key fields.

The multi-purpose DataStore Object (advanced) supports many different modeling scenarios across many different EDW layers. The various DataStore Object (advanced) use cases are implemented by choosing relevant settings in the definition of the DataStore Object (advanced).

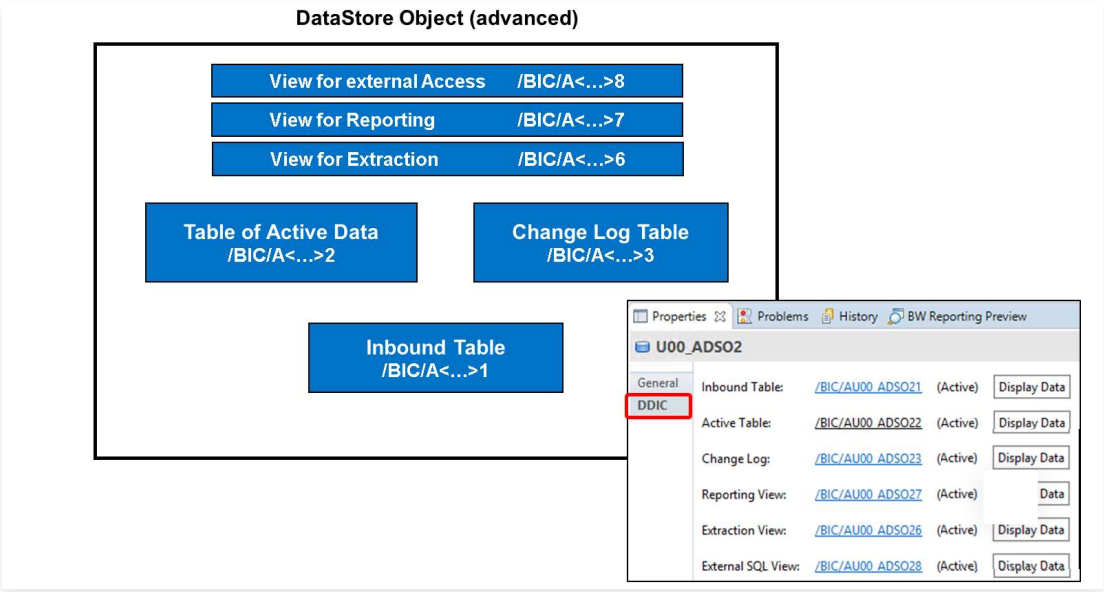
In SAP BW/4HANA, there are four types of DataStore Object (advanced):

- Standard - for generic use
- Data Mart - to develop business-specific data sets
- Staging - to manage inbound data that needs harmonizing and combining with other data sources
- Direct Update - to bypass the standard loading process

(advanced) during the loading process.

The DataStore Object (advanced) is now created and maintained in the BW Modeling Tools. However, the management of the data loading to a DataStore Object (advanced) is supported with SAP BW/4HANA Cockpit.

Structure of the DataStore Object (advanced)



As with SAP BW, the DataStore Object (advanced) in SAP BW/4HANA continues to be technically organized around three core tables (Inbound Table, Table of Active Data, and Change Log). These tables are generated when the DataStore Object (advanced) is created and activated. The use of the tables depends on the modeling options that have been selected. Regardless of the modeling option choices, the three tables are always generated to support changes to the use case of the data model, without disrupting the underlying database.

With the exception of "Direct Update", data is always loaded into the inbound table first. Then, data is either read from this table directly, or it is processed further into the other one or two tables for reading or further extraction. This depends on the setting you have chosen when you created the DataStore Object (advanced).

Let's explore the three core tables:

- Structure: Request ID (REQTSN), Data Package (DATAPAKID), Record number (RECORD), Record mode (RECORDMODE), Key Field 1, Key Field n, Field 1, Field n...
- Key Definition: REQTSN / DATAPAKID / RECORD (= generated technical key)

- **Table of Active data: /BIC/A<technical name>2**
  - Equals active table of the DataStore Object (classic) or compressed fact table of a non-HANA-optimized InfoCube in the past
  - Structure: Key Field 1, Key Field n, Record mode (RECORDMODE), Field 1, Field n...
  - Key Definition: Key Field 1 / Key Field n (= user defined semantic key)
- **Change Log table: /BIC/A<technical name>3**
  - Same as for DataStoreObject (classic) in the past
  - Structure: Request ID (REQTSN), Data Package (DATAPAKID), Record number (RECORD), Record mode (RECORDMODE), Key Field 1, Key Field n, Field 1, Field n....
  - Key Definition: REQTSN / DATAPAKID / RECORD (= generated technical key)

In addition to these three core tables, the following three table views are created as well. Which of the three core tables they refer to depends on the modeling scenario (that is, the setting of the modeling properties):

- **View for Extraction** from the DataStore Object (advanced): **/BIC/A<technical name>6**
- **View for Reporting** on the DataStore Object (advanced): **/BIC/A<technical name>7**
- **View for external SQL Access** to the DataStore Object (advanced): **/BIC/A<technical name>8**

SAP does not support the scenario where a customer uses code to directly access any of the three tables storing DataStore Object (advanced) data (refer to SAP Note 1682131 for the explanation). To support those customers who would like to do this, SAP BW/4HANA provides a new database view for external SQL access. It is a stable interface that allows official and authorized

native SAP HANA calculation view provides.

The External SAP HANA SQL View is automatically generated during activation of an DataStore Object (advanced). It is created in the ABAP Dictionary and therefore also in the database catalog for use with native SQL and OpenSQL/ABAP. For DataStore Objects (advanced) that were activated prior to the introduction of the External SAP HANA SQL View feature, you can create the missing views using report *RSDG_ADSO_ACTIVATE*.

The *Repair* button, with *Rebuild database views* selected, only creates the view(s) in the database catalog, but not in the ABAP Dictionary. To create the view(s) in the ABAP dictionary, you have to activate the DataStore Object (advanced) manually using the BW Modeling Tools or the report. The activation must not take place in the ABAP Dictionary directly.

If the data model is inventory-enabled to manage noncumulative key figures, then the following tables are also created:

- **Validity Table: /BIC/A<technical name>4**
- **Reference Point Table: /BIC/A<technical name>5**

To support data temperature management, you can perform exceptional updates to the external cold store using the generated view: **/BIC/A<technical name>9** .

Launch the next demo to explore the settings of a DataStore Object (advanced) in BW Modeling Tools.

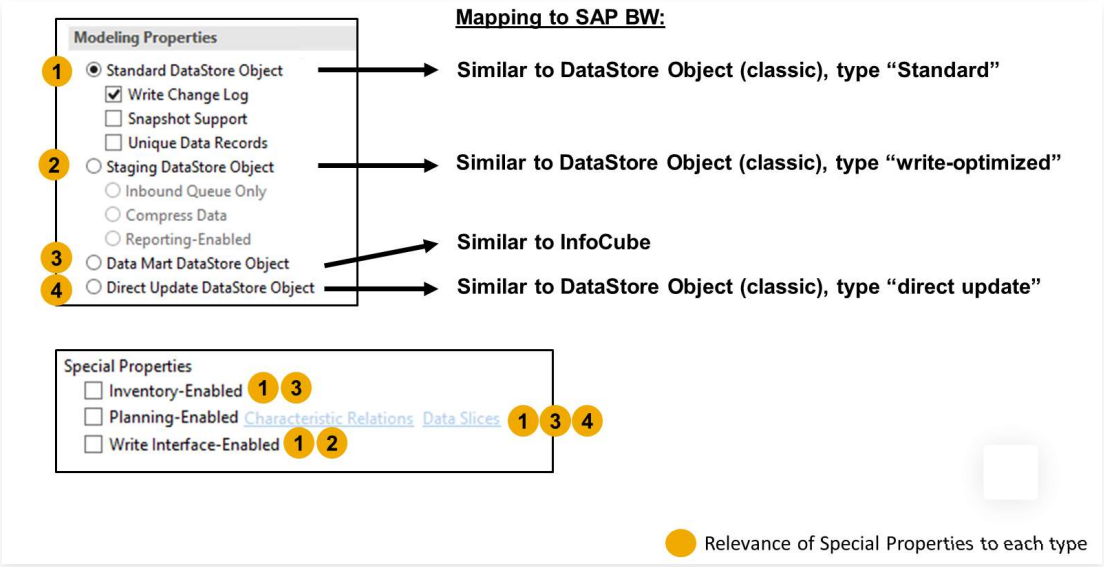
Explore a DataStore Object (advanced) with BW Modeling Tools

Demo

Modeling Scenarios

Core Modeling Properties

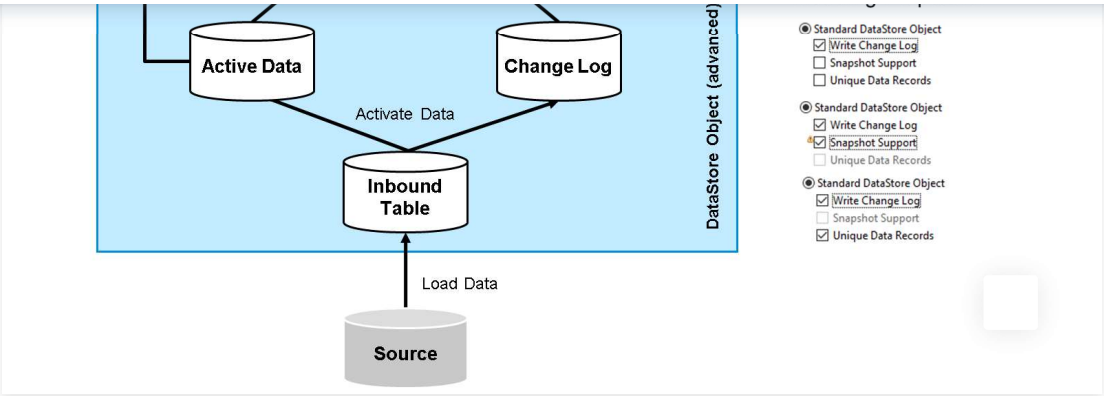
In SAP BW/4HANA, the DataStore Object (advanced) is the central object storing and consolidating transactional data. The DataStore Object (advanced) can be used in many different ways, depending on the core modeling properties you select.



The core modeling properties, located in the *General* tab in the BW Modeling Tools, represent the key settings that define the behavior of the DataStore Object (advanced) and the usage of its three database tables. There are also some additional enhancements called "special properties" regarding inventory key figures, planning functions, and a new write interface, which are available depending on the modeling scenario you choose.

Standard DataStore Object

Let's start with one of the most popular types of DataStore Object (advanced).

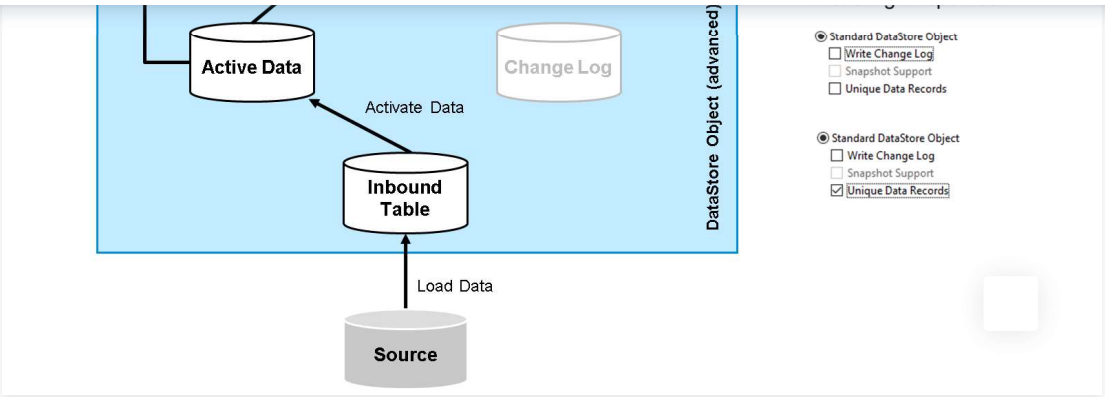


To model a DataStore Object (advanced) as a standard type that uses a Change Log, choose *Standard Data Store Object* and the setting *Write Change Log*. With a Change Log the delta (new, deleted, and changed records) is saved in the Change Log. The Change Log is used to extract the delta. Requests can only be rolled back from the DataStore Object (advanced) if there is a Change Log. That is, the status before the activation of the request can be restored. A DataStore Object (advanced) of the type standard with only the option *Write Change Log* selected can be compared with the former Standard DSO (classic).

The option *Snapshot Support* can be used if your DataSource only delivers the current dataset as *FULL* and not delta. By setting this indicator, deleted data records are 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.

You can use the option *Unique Data Records* if you only load unique data records (data records with nonrecurring key combinations) into the DataStore Object (advanced). During activation of a request, the system checks whether unique records exist. If a record already exists, the activation is canceled with an error. This option is useful to speed up the loading process since only inserts are performed during loading.

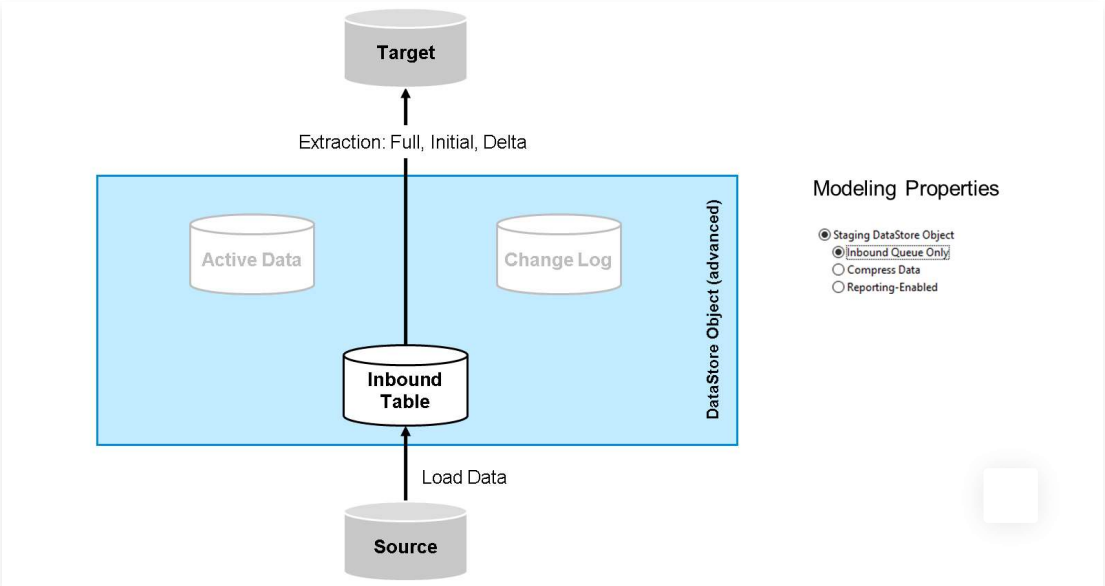
You can save system resources if you do not require a delta extraction from this source object.



Simply do not request a Change Log. Only the Inbound Table and the Active Table are used. Also, in this scenario, it is possible to choose *Unique Data Records* setting.

Staging DataStore Object

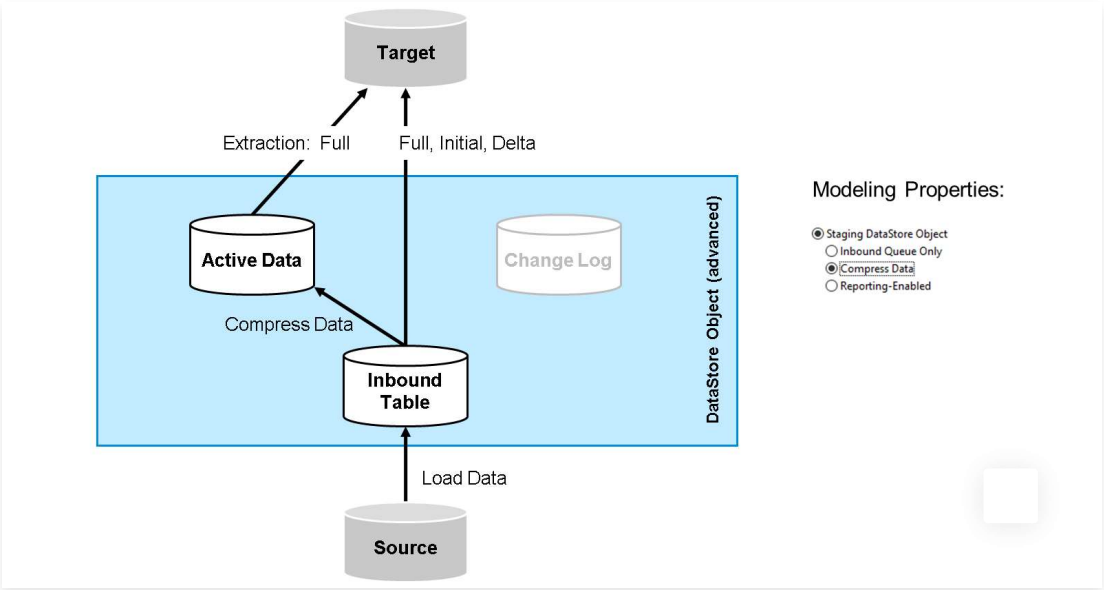
DataStore Objects (advanced) of the type Staging are optimized for very fast loading. These are usually found in the inbound layers of the EDW architecture to ensure that data is transferred from source systems as quickly as possible.



When this type is chosen, there are additional settings to fine-tune the model.

For example, when you set the option *Inbound Queue Only*, no data is saved in the Change Log. The extraction process always reads the data in the

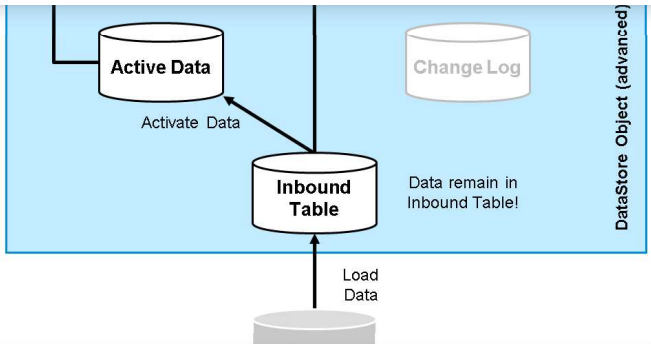
know the activation logic and therefore cannot aggregate over records.

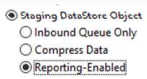


When you set the option *Compress Data,* the data of a request is written to the inbound table. During the compression process, the data is written to the active table (aggregated in accordance with the semantic key of the active data table) and deleted from the inbound table. To save memory space, the change log is not filled. Therefore, you cannot perform request-based deletion of data from the DataStore Object (advanced). You can only delete data selectively.

For full extraction, the inbound table and the table with the active data are accessed. Delta extraction only uses the inbound table.

A DataStore Object (advanced) with the property *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.





data is written to the
the inbound table. Since
he data can be deleted
ble. The data is only
es place on the table
nly visible after

art is optimized for
P BW .

following features:

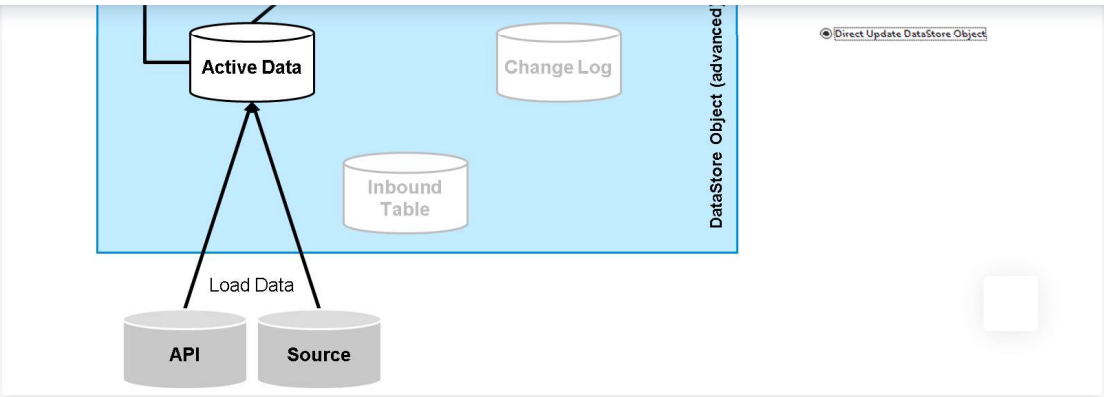
and not overwritten.

- *Loading*: The data is always loaded into the inbound table.
- *Activation*: When you activate the data, the data is copied to the table of active data and grouped according to the aggregation behavior. Afterward, the data is deleted from the inbound table. The change log is not filled. The activation compares to a compression of InfoCubes of SAP BW.
- *Rollback*: For this type, requests can only be deleted if the request has not yet been activated.
- *Extraction*: Full and initial extraction read the data for the update into a further data target as a union of the inbound table and the table of active data.

  The data for the delta extraction is read from the inbound table only.
- *Reporting*: The query reads data as a union 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 (advanced) does not change during the navigation steps in the query.

Direct Update DataStore Object

This type of The DataStore Object (advanced) was available in SAP BW.

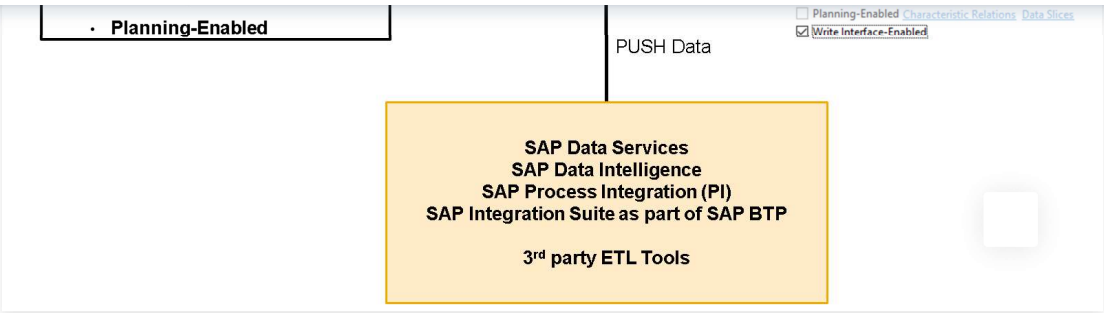


With the DataStore Object (advanced) for direct update, you can load data directly to the table of active data, including standard consistency checks (for example, SID processing, consistency of time characteristics, areas locked by cold store, or NLS).

Data can be loaded by standard DTP/Transformation or by the following APIs:

- RSDSO_DU_WRITE_API / RSDSO_DU_WRITE_API_RFC: Loads data from an internal table to the table of active data.
- RSDSO_DU_DELETE_API_RFC: Deletes data from the table of active data. This one can be truncated or selectively deleted.
- RSDSO_DU_CLEANUP_API_RFC: Deletes API requests with errors. Red requests block further load requests by DTP/Transformation or by API.

Write Interface



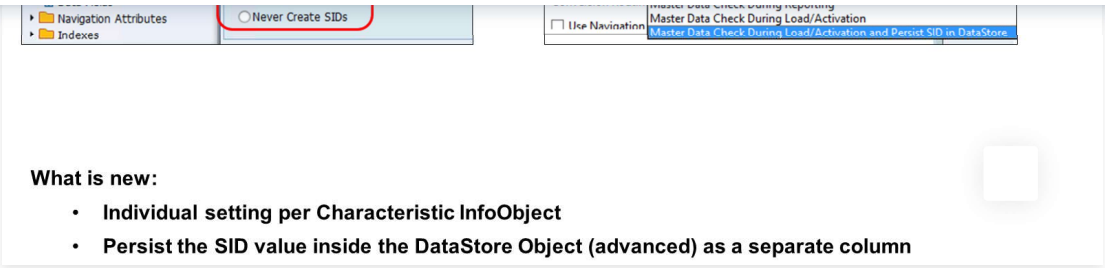
When you use the option *Write Interface-Enabled*, the data in this DataStore Object (advanced) can be moved to the inbound table using SAP integration solutions by PUSH. Based on this property, it is the successor for the obsolete BW source system types Data Services, Web Service and External System. The interface can be integrated with the following solutions:

- SAP Data Services
- SAP Data Intelligence
- SAP Process Integration (PI)
- SAP Integration Suite as part of the Business Technology Platform (BTP), formerly also known as SAP Cloud Platform Integration (CPI)
- 3rd party ETL Tools

The option *Write Interface* can be used for all types of Standard DataStore objects and Staging DataStore objects, but not together with *Inventory-Enabled* or *Planning-Enabled*.

SID Generation Settings

The basic concept of the Surrogate ID (SID) has not changed with SAP BW/4HANA. SIDs are still required for reporting based on BW Queries. But the generation options have been improved.



In SAP BW, the label is *BEx-Flag* or *SID Generation*. In SAP BW/4HANA it is called *Master Data Check*.

Compared to the DataStore Object (classic), here are the differences compared to the SID generation for a DataStore Object (advanced):

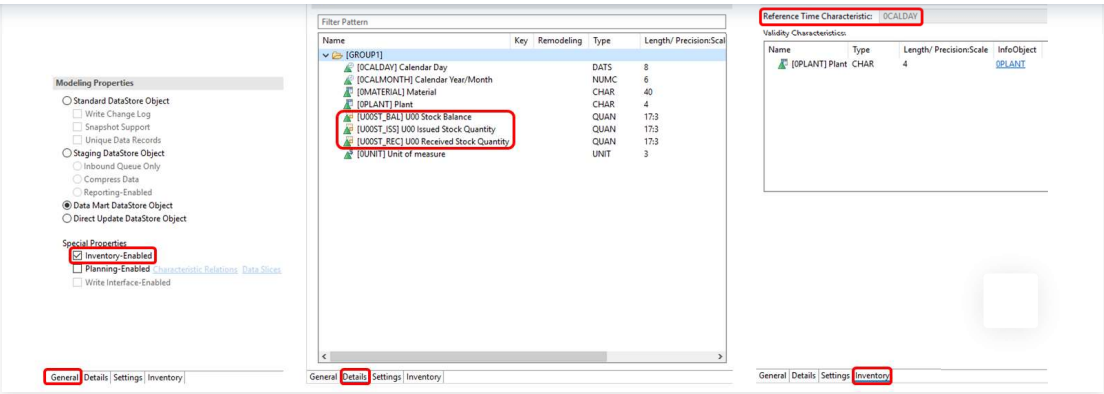
- In the past there was one general setting for the whole DataStore Object. Now there is an individual setting per Characteristic InfoObject.
- There is a new additional option to persist the SID value in the DataStore Object (advanced) as a separate column.

In a DataStore Object (advanced), usually only the characteristic values are saved, and not the corresponding SIDs. In this case an SQL Join is executed for each characteristic to its corresponding SID table at query time. This has a negative effect on the performance. For InfoObjects that are used frequently in reporting and which have a high cardinality, the new option is recommended. You can use the report SAP_ADSO_DESIGNS to determine whether this might be helpful for your model (SAP Note 2511639).

Hint

For more details, refer to this SAP blog post: https://community.sap.com/t5/technology-blogs-by-sap/demystifying-sap-bw-adso-master-data-check/ba-p/13505427 .

Modeling Inventory Scenarios with Noncumulative Key Figures



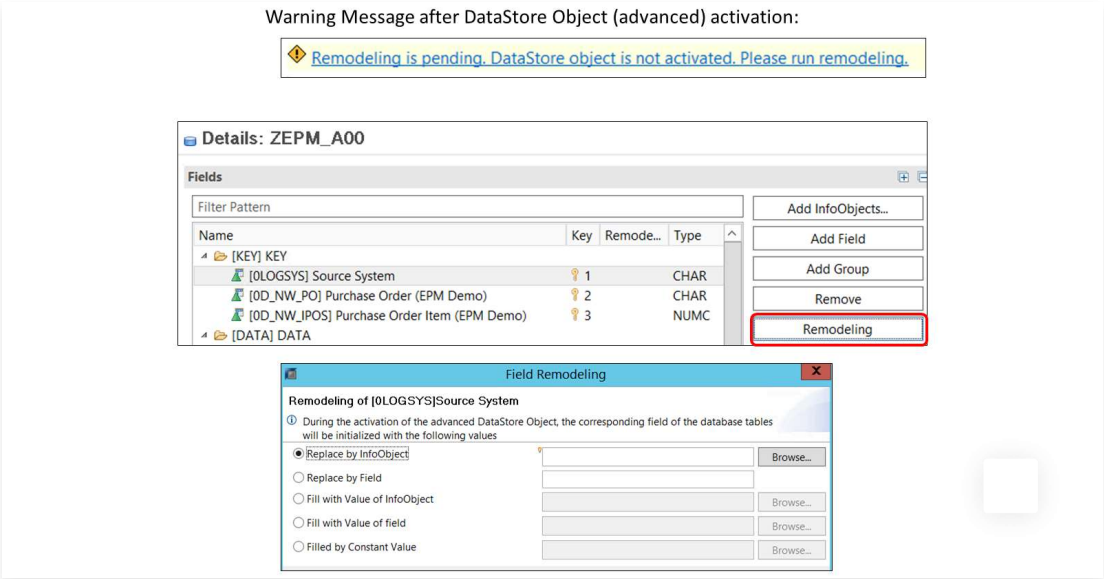
In SAP BW/4HANA, you can use the DataStore Object (advanced) to model noncumulative key figures.

- Precondition: The *Inventory-Enabled* flag in the core modeling properties is switched on.
- If you add a noncumulative key figure InfoObject to the data model, the system also displays the *Inventory* tab. When the noncumulative key figure is added, the key figures for noncumulative changes (that is, for inflows and outflows), are also added to the DataStore Object (advanced).
- In this case, two additional tables are created: `/BIC/A<techn.name>4` = Validity table and `/BIC/A<techn.name>5` = Reference Point table.
- The reference points required for calculation of the noncumulative key figures are updated with the change log data. The data model must therefore have a change log, and a key must also have been defined. The following properties are required: (1) Activate/Compress Data and Write Change Log or (2) Activate/Compress Data and All Characteristics are Key, Reporting on Union of Inbound and Active Table.
- It is possible to delete reference points without corresponding movements based on tr. `DELETE_FACTS` .
- InfoObjects must be used as key figures, noncumulative key figures, and validity characteristics. Fields are not supported for this scenario.

Hint

For additional details, refer to the following source: SAP First Guidance - SAP BW/4HANA: Inventory Handling and Non-

Remodeling DataStore Object (advanced)



If your DataStore Object (advanced) is empty, changes are possible in the BW Modeling Tools without further activities. However, if it already contains data, the system follows a predefined logic to decide whether an additional manual remodeling task is required in order to activate the object properly. This also applies to the transport management when you provision these changes to the production system finally. The separate remodeling job ´s role is to keep the activation time to a minimum and thus to avoid terminations during the direct execution (for example, time-outs). This safety belt is justified, because changes to DataStore Object (advanced) with a lot of data might result in a huge amount of read and write operations to the existing data in the DataStore Object (advanced).

The remodeling requirements are not limited to the changes offered by the *Remodeling* button in the DataStore Object (advanced) modeling UI of the BW Modeling Tools. The remodeling framework is also used when other changes require crucial read and/or write activities on the DataStore Object (advanced) database tables. For example changes to general modeling properties, changes to the DataStore Object (advanced) key, changes to indices, partitions or Data Tiering settings.

of working:

- Replace InfoObject by another one
- Replace InfoObject by a field
- Replace field by another one
- Replace field by an InfoObject
- Fill with value of InfoObject
- Fill with value of field
- Fill with constant value

2. During activation, the system checks whether the object can be activated immediately, or whether it needs to be remodeled first. If remodeling is required, a warning appears: `Remodeling is pending. DataStore object is not activated. Please run remodeling` . This means, the object is not activated yet and a remodeling request is created. If you have made two or more changes to your object, these changes are compiled into one remodeling request. You can continue to use the old DataStore Object (advanced) version. Until that, the remodeling request has not been executed.
3. Open the app *Remodeling Requests* in the SAP BW/4HANA Cockpit or use transaction `RSMONITOR` to open the remodeling monitor in your SAP BW/4HANA system. Identify the generated request and start or schedule it manually.
4. If you want to transport a remodeled DataStore Object (advanced) to the test system and finally to the productive system, and it already exists in the productive system, the system checks whether remodeling is necessary, and the required remodeling request is created in the target system. If you want to transport an DataStore Object (advanced) for which a remodeling request has been created but not yet executed, a warning informs you that the M version and A version are different. You can only transport the A version (without the current changes).

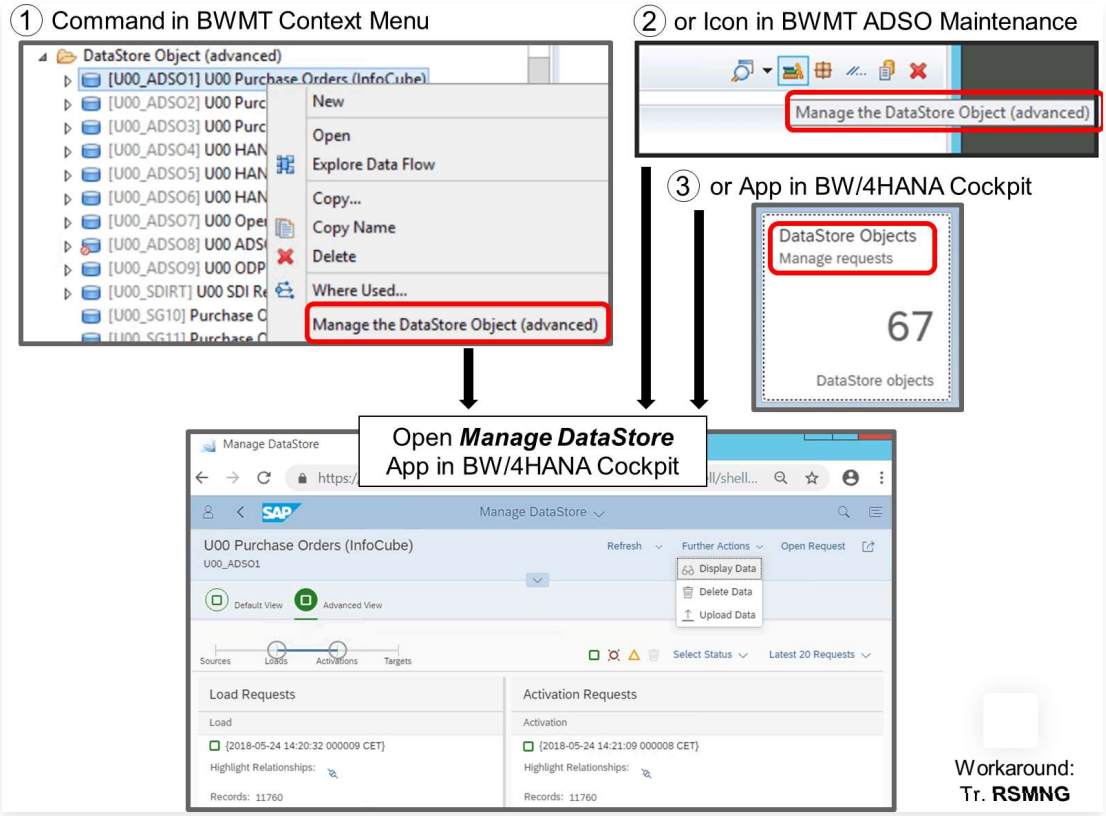
Hint

For more details, refer to the blog *Role of Remodeling in the ADSO Change Management*

Managing Data Loads

Managing the DataStore Object (advanced)

Data load management of the DataStore Object (advanced) is provided through an app in the SAP BW/4HANA Cockpit. With this app you can display loads requests, display loaded data, delete requests, identify the activation status of loads and more.



The *Manage* function of DataStore Object (advanced) can be launched directly from the BW Modeling Tools in Eclipse. Launching from BW Modeling Tools in Eclipse opens a new web browser and automatically logs on to the SAP BW/4HANA Cockpit where it directly opens the *DataStore Objects - Manage Requests* app. The following functions are available:

- Request management of loading and activation processes including their logs

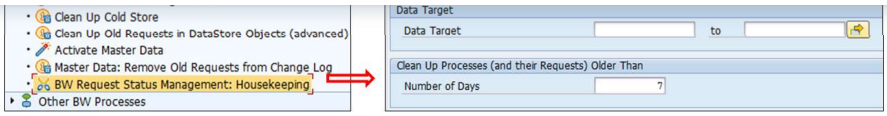
- Upload data from a flat file

New Request Management

Request management is central to SAP BW/4HANA's managed approach to data warehousing. The increase of the SAP BW systems, the greater complexity and the demand for more real-time scenarios caused SAP to rewrite the existing request management and process management (**RSSM**). As a consequence, request IDs based on 0REQUID are not available with SAP BW/4HANA anymore.

Instead, SAP BW/4HANA has a new request status and process management (**RSPM**) based on **Request Transaction Sequence Numbers (TSN)** based on 0REQTSN. RSPM is used in data staging, distribution, streaming, and planning processes for InfoObjects, DataStore Objects (advanced), and Open Hub Destinations. The new request management comes with a new **manage UI** that is directly accessible from the SAP BW/4HANA Cockpit. It enables you to quickly navigate through very large sets of requests and protocols and to perform manual tasks or monitoring activities.

In RSPM, the TSN is no longer an INT4 value, but a timestamp plus a generated, increasing postfix (for example, *2019-03-13 10:25:11 000004*). Not only does it remove the 2 billion limitation for the system-wide number of requests, it also allows for new semantics derived directly out of the request ID, that is, the date and time of loading. It has a conversion exit to display the time stamp in a local time zone. Example: 99991231235959.000009900 (internal view of TSN), {9999-12-31 23:59:59 0000099 CET} (user view of TSN). The Request TSN has 2 navigational attributes: User (0AUSER) and Source of Data (0ASOURCE) which are populated at creation time of the Request TSN and available as meta data in BW queries.



RSPM Housekeeping is a recommended recurring task. Why?

1. Improve Data processing in your system (loading, activation, delta determination)
2. Make sure InfoProvider administration is fast and easy (due to shrinked request history)
3. Avoid the RSPM* BW/4HANA backend tables to grow constantly

RSPM Housekeeping aggregates and archives the RSPM request details into RSPMHK*-Tables.

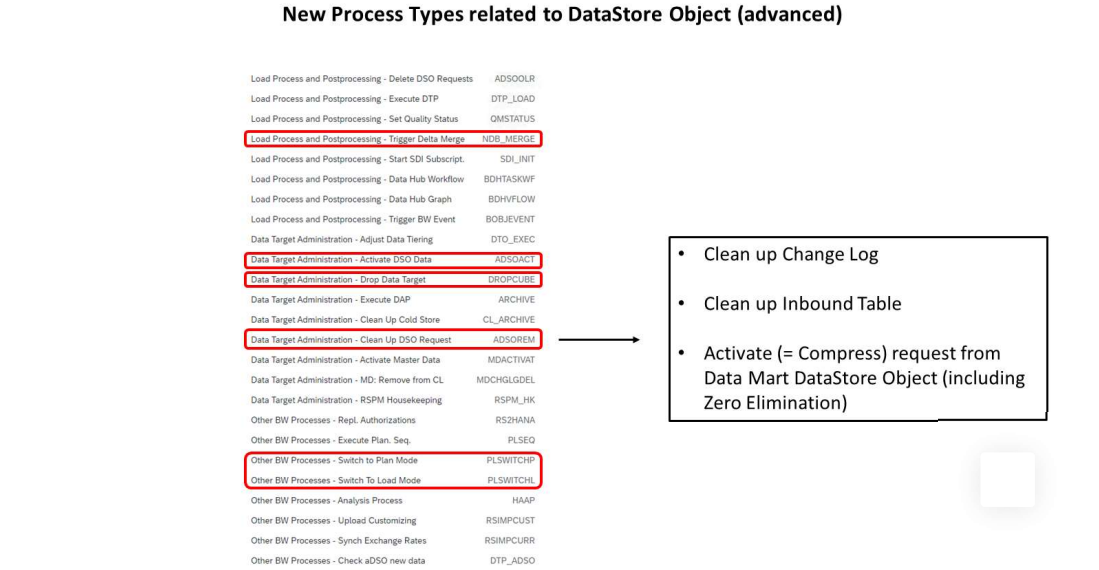
The following requests can be archived:

- All deleted requests
- All requests from InfoObjects, Open Hub Destinations, or ADSOs of type direct update
- All requests from the inbound queue of ADSOs of type Data Mart or Standard, if the request has been activated ("compressed")
- All requests from ADSOs of type Standard with Change Log, if the request has been provisioned to all data targets and if the request was deleted from the Change Log afterward

Without recurring clean-up and housekeeping, RSPM request details would grow considerable in an SAP BW/4HANA system. To perform this task, SAP delivered a new process variant, which archives those request information, which is not required anymore. For example, if a request is removed from the InfoProvider, the related request details can also be removed. More details can be found in this blog post:

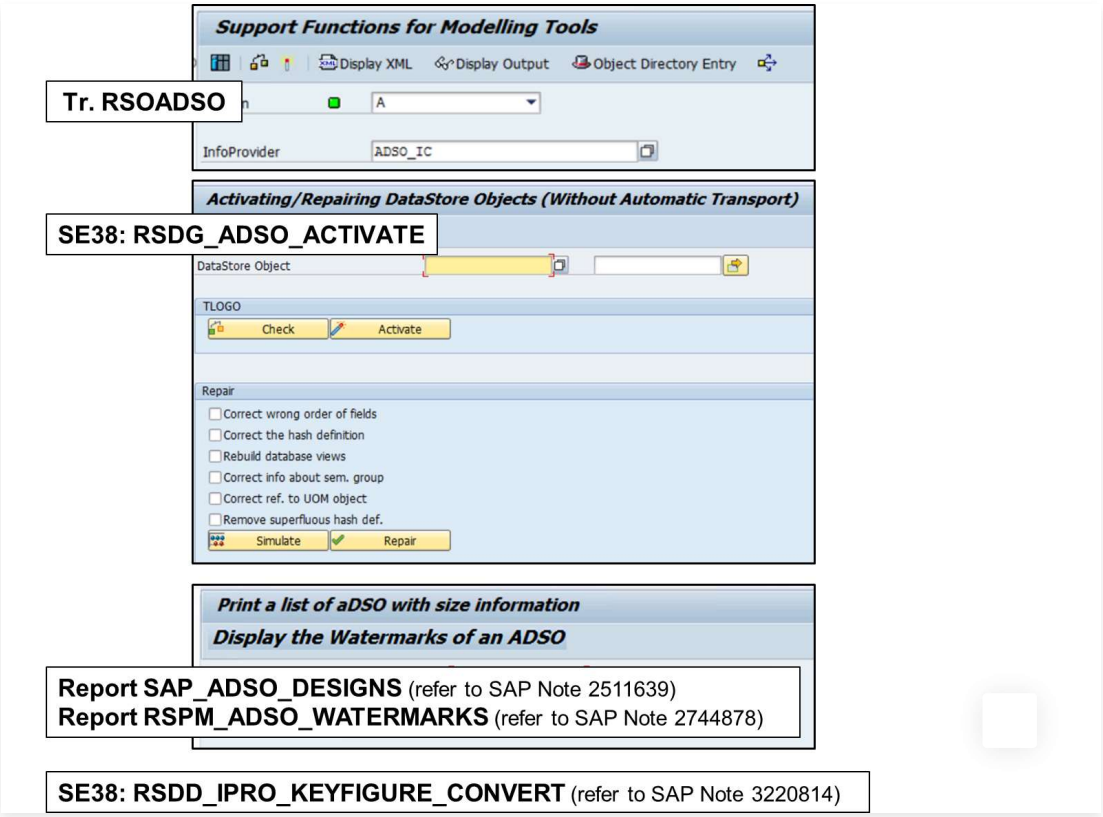
https://community.sap.com/t5/technology-blogs-by-sap/getting-started-with-sap-bw-request-housekeeping/ba-p/13640732

Managing DataStore Object (advanced) with Process Chain Types



- Process type **Activate DSO Data** is available to activate requests from the DataStore Object (advanced).
- Process type **Drop Data Target** is available to delete the complete contents from the DataStore Object (advanced).
- Process type **Clean Up DSO Request** is available to remove requests from the inbound table or change log. For DataStore Object (advanced) of type Data Mart it can be used to activate requests, which means compression including Zero Elimination.
- Process types **Switch to Plan Mode** and **Switch to Load Mode** have been added to set up planning scenarios.

Helpful Tools to Support DataStore Object (advanced) administration



There are several additional tools to analyze the DataStore Object (advanced) and repair their inconsistency:

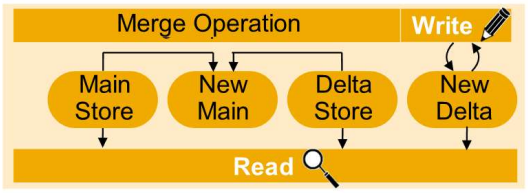
- Report *RSDU_ADSO_ACTIVATE* to activate these objects and repair certain issues in their data model definition
- Report *SAP_ADSO_DESIGNS* to analyze the size of a DataStore Object (advanced) and plan housekeeping activities
- Report *RSDD_IPRO_KEYFIGURE_CONVERT* converts amounts into a specified target currency based on SAP Standard TCUR logic. The tool is delivered by SAP note 3220814 and provides a detailed documentation in its attachment. Corrections and enhancements are provided on the same component (BW-BEX-OT-DBIF). Currently there is patch 1 (3245254), patch 2 (3251668), and patch 3 (3259972).

Understanding DataStore Object (advanced) Delta Merge Settings

Column-store tables are good at reading performance, but poor at writing or receiving new, updated data. For this reason, in-memory column-store tables consist of a main storage (read-optimized), as well as a small delta storage (write-optimized). When reading such a table, both the main and delta parts are relevant. This split is fully transparent for the application and its users, and is normally managed by the SAP HANA database. The same is true for the delta merge process. The transfer of data from delta to main is normally managed by the database, and is a task performed by the database administrator.

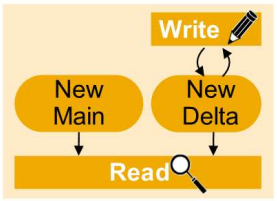
2. During delta merge operation:

- Read/write access still works on old stores and new delta store



3. After delta merge operation:

- Data optimized for fast access again

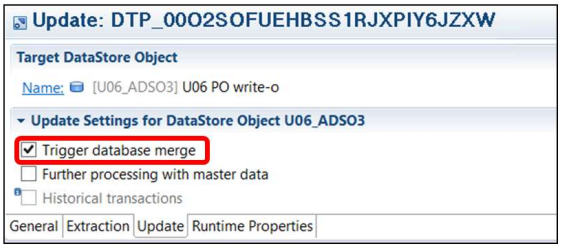


A delta merge is used to transfer modifications from the delta storage to the main storage. First, an asynchronous check is performed to see whether a delta merge is required. If a threshold value is exceeded, the merge is carried out in the delta storage. When a read access is executed, the data is read from the main storage and the delta storage, and the results are merged.

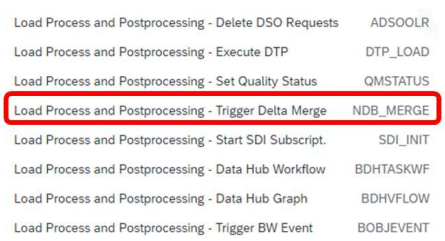
For DataStore Object (advanced) the Delta Merge must be triggered by SAP BW/4HANA.

Options:

1. **In DTP**



2. **In Process Chain**



Because SAP BW/4HANA normally processes mass data in DataStore Object (advanced), the delta merge is managed by the application in this case and

DTP request has been successfully processed, this setting controls the delta merge. The checkbox is selected by default.

2. In exceptional cases, performing a delta merge after processing a DTP request is not recommended because of load balancing issues. In these situations, the process type **Trigger Delta Merge** can be used within a process chain to execute the delta merge.

Make sure that the delta merge is triggered by either the DTP or the process type. Otherwise, data remains in the delta storage table. Over time, this leads to suboptimal memory use and read performance. Note that the limit of two billion rows per partition also applies to the SAP HANA delta storage.

Managing Large Data Volume

Introduction

In SAP BW/4HANA, the DataStore Object (advanced) is the recommended data model to manage transactional data. Special considerations are required, when the planned data volume is very large and expected data growth is considerable. If you do not take steps to manage data growth, you can expect the performance of data loading and reporting to be negatively affected.

SAP provides solutions to handle large volumes of data including:

- **Indexes**: Additional indexes can be created by the BW developer in the DataStore Object (advanced) modeling UI. This does not reduce the data volume, of course. But it is a valid option to provide fast read access to mass data.
- **Database Partitioning**: Database tables created for a DataStore Object (advanced) and InfoObjects with property 'high cardinality' are partitioned on the SAP HANA database by default. For DataStore Object (advanced), you can maintain additional options for database partitioning in the BW Modeling Tools.
- **Semantic Partitioning**: To address some of the aspects of dealing with big data in SAP BW/4HANA systems, it is recommended to partition a DataStore Object (advanced) semantically, that is, model them as Semantic Groups.

Housekeeping: If data is no longer needed it is recommended to delete it before it reaches critical limits. In general, the same logic and the same tools apply as in the past with SAP BW.

Indexes

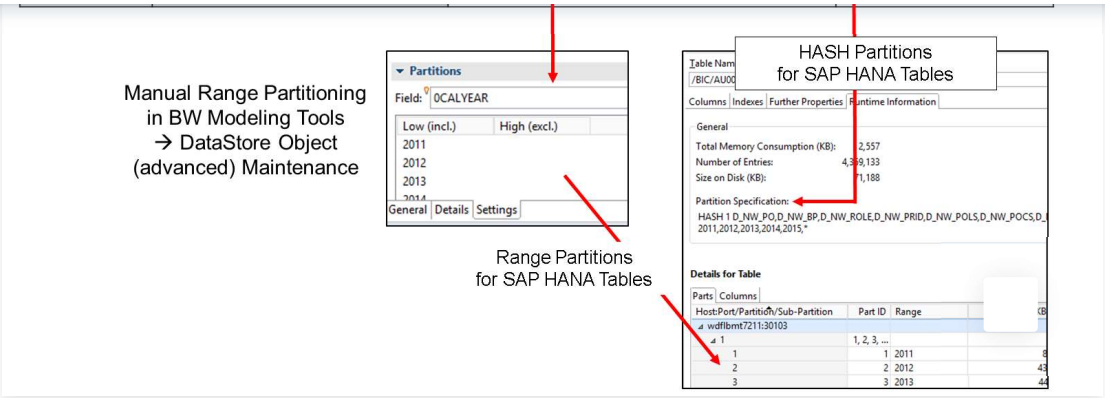
Standard primary indexes are created for the DataStore Object (advanced) by default. They are required for reading DataStore Object (advanced) data, or for compounded characteristics, to accelerate the join between the master data tables and the DataStore Object (advanced) tables during reporting (for example, query processing).

You can create additional secondary indexes for a DataStore Object (advanced), if necessary (for example, for a very complex lookup which does not meet the performance expectations). These customer indexes are created on the active table or, if no active table is found, the index is created on the inbound table. Take into account the resource requirements that an index requires in terms of SAP HANA memory and index update processing; use indexes only if performance expectations cannot be met without them.

> **Note**
>
> For more details, refer to SAP Note 2160391 (FAQ: SAP HANA Indexes).

Database Partitioning

SAP HANA tables can store up to 2 billion records. This is usually sufficient for most organizations. However, if we partition the tables then the 2 billion limit record applies to each database table partition. This way, you can manage several billion records of data in one single table, for example, the DataStore Object (advanced) active table.


Manual Range Partitioning
in BW Modeling Tools
→ DataStore Object
(advanced) Maintenance
Partitions
Field: 0CALYEAR
Low (incl.)
High (excl.)
2011
2012
2013
General
Details
Settings
Range Partitions
for SAP HANA Tables
HASH Partitions
for SAP HANA Tables
Columns
Indexes
Further Properties
General
Total Memory Consumption (KB):
Number of Entries:
Size on Disk (KB):
Partition Specification:
Details for Table
Parts
Columns
Host:Port/Partition/Sub-Partition
Part ID
Range

partition values, you can also perform the same in a **dynamic** way. In the case of this so-called dynamic partitioning, you only specify the partition field but not the explicit value. The partitions are then created dynamically during activation of the data. Dynamic partitioning is therefore more flexible than static partitioning.

Since a new partition is created for every new value of the partition field when data is activated, a large number of partitions might be created which results in a negative effect on performance. **As a rule of thumb, the number of partitions in the DataStore Object (advanced) active table should not exceed 50** (HASH partitions x RANGE partitions). Dynamic partitioning is therefore only suitable for characteristics with low cardinality. You can reduce the number of dynamic partitions by leveraging a SAP time characteristic (0CAL*, 0FISC*) as partition field, or a derived characteristic or a field with data type DATS as partition field. You can define a partition granularity for these fields, such as calendar year (0CALYEAR) for partition field calendar month (0CALMONTH). In this example, all values belonging to a calendar year are grouped into a range partition, and fewer partitions are created than there would have been without granularity.

2. Rule of Thumb 2: Upper limit of 50 partitions (Number of primary partitions x Number of secondary partitions)
3. Rule of Thumb 3: Lower Limit of 50 million records per each DataStore Object (advanced) RANGE-partition

Best Practices for manual modeling of DataStore Object (advanced) RANGE partitions:

1. The partitioning object must be a key InfoObject.
2. These partitions are only defined for the DataStore Object (advanced) table of active data.
   → This means, DataStore Object (advanced) of type "Data Mart" need to be activated ("compressed") to benefit from it.
3. Recommended partitioning objects are the standard SAP Time Characteristics (0CAL*/0FISC*), because they can leverage pruning by default including the related time hierarchies: This means the systems knows in advance which partition(s) to read. (e.g. Partitions per year & Query access by month(s): During runtime, year is derived from month → the relevant partition(s) are identified properly)
4. There is no difference between partition definition on single values or ranges.
   (for example, 0CALMONTH = JAN 2022 equals 0CALDAY = JAN 01, 2022 - JAN 31, 2022)
5. The BW Query definition should contain the partitioning object as a filter if possible.
6. Consider that these partitions are also crucial elements for Data Tiering Optimization (DTO).

Hint

More information about large volumes and partitioning can be found in this documentation: https://help.sap.com/docs/SUPPORT_CONTENT/bwplaolap/3361386590.html In addition refer to SAP Note 2374652 (Handling Very Large Data Volumes in SAP BW/4HANA).

Next lesson

Was this lesson helpful? Yes No