

Table of Contents

| | |
|---|----|
| DataStore Object | 3 |
| Definition | 3 |
| Use | 3 |
| DataStore Objects for Direct Update | 4 |
| Definition | 4 |
| Use | 4 |
| Structure..... | 4 |
| Application programming interface (API) | 5 |
| Analysis Process Designer | 6 |
| Use | 6 |
| Integration | 6 |
| Restrictions | 7 |
| Functions..... | 7 |
| Versioning | 8 |
| Creating DataStore Objects for Direct Update | 8 |
| Integration | 8 |
| Designing a Direct-Update DSO | 9 |
| Settings in Direct Update DSO | 12 |
| Type of DataStore Object..... | 12 |
| Including Key Fields and Data Fields in the DSO | 12 |
| Final Steps | 13 |
| Related Content | 14 |
| Disclaimer and Liability Notice | 15 |

DataStore Object

Definition

A DataStore object serves as a storage location for consolidated and cleansed transaction data or master data on a document (atomic) level.

This data can be evaluated using a BEx query.

A DataStore object contains key fields (such as document number, document item) and data fields that, in addition to key figures, can also contain character fields (such as order status, customer). The data from a DataStore object can be updated with a delta update into InfoCubes (standard) and/or other DataStore objects or master data tables (attributes or texts) in the same system or across different systems.

Unlike multidimensional data storage using InfoCubes, the data in DataStore objects is stored in transparent, flat database tables. The system does not create fact tables or dimension tables.

Use

Overview of DataStore Object Types

| Type | Structure | Data Supply | SID Generation |
|-------------------------------------|--|----------------------------|----------------|
| Standard DataStore Object | Consists of three tables: activation queue, table of active data, change log | From data transfer process | Yes |
| Write-Optimized DataStore Objects | Consists of the table of active data only | From data transfer process | No |
| DataStore Objects for Direct Update | Consists of the table of active data only | From APIs | No |

DataStore Objects for Direct Update

Definition

The DataStore object for direct update differs from the standard DataStore object in how the data is processed. In a standard DataStore object, data is stored in different versions (active, delta, modified), whereas a DataStore object for direct update contains the data in a single version. Therefore, data is stored in precisely the same form in which it was written to the DataStore object for direct update by the application. In the BI system, you can use a DataStore object for direct update as a data target for an analysis process. For more information, see

The DataStore object for direct update is also required by diverse applications, such as SAP Strategic Enterprise Management (SEM) for example, as well as other external applications.

Use

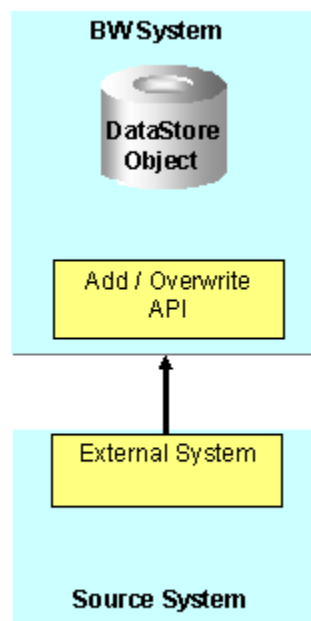
DataStore objects for direct update ensure that the data is available quickly. The data from this kind of DataStore object is accessed transactionally, that is, data is written to the DataStore object (possibly by several users at the same time) and reread as soon as possible.

It offers no replacement for the standard DataStore object. Instead, an additional function displays those that can be used for special applications.

Structure

The DataStore object for direct update only comprises a table for active data. It retrieves its data from external systems via fill- or delete- APIs.

The load process is not supported by the BI system. The advantage to the way it is structured is that data is easy to access. They are made available for reporting immediately after being loaded.



Application programming interface (API)

It is important to understand the concept of APIs before understanding a Direct Update DSO.

An application programming interface (API) is a particular set of rules and specifications that software programs can follow to communicate with each other. It serves as an interface between different software programs and facilitates their interaction, similar to the way the user interface facilitates interaction between humans and computers.

An API can be created for applications, libraries, operating systems, etc., as a way of defining their "vocabularies" and resources request conventions (e.g. function-calling conventions). It may include specifications for routines, data structures, object classes, and protocols used to communicate between the consumer program and the implementer program of the API.

The DataStore object for direct update consists of a table for active data only. It retrieves its data from external systems via fill or delete APIs.

The following APIs exist:

- RSDRI_ODSO_INSERT: Inserts new data (with keys not yet in the system).
- RSDRI_ODSO_INSERT_RFC: see above, can be called up remotely
- RSDRI_ODSO_MODIFY: inserts data having new keys; for data with keys already in the system, the data is changed.
- RSDRI_ODSO_MODIFY_RFC: see above, can be called up remotely
- RSDRI_ODSO_UPDATE: changes data with keys in the system
- RSDRI_ODSO_UPDATE_RFC: see above, can be called up remotely
- RSDRI_ODSO_DELETE_RFC: deletes data

Analysis Process Designer

Use

In SAP BW, data from various databases from systems available in the company are collected, consolidated, managed and prepared for evaluation purposes. There is often further, valuable potential in this data.

It deals with completely new information that is displayed in the form of meaningful connectivity between data but that is too well hidden or complex to be discovered by simple observation or intuition.

The Analysis Process Designer (APD) makes it possible to find and identify these hidden or complex relationships between data in a simple way. Various data transformations are provided for this purpose, such as statistical and mathematical calculations, and data cleansing or structuring processes.

The analysis results are saved in BW data targets or in a CRM system. They are available for all decision and application processes and thus can be decisive (strategically, tactically, and operatively).

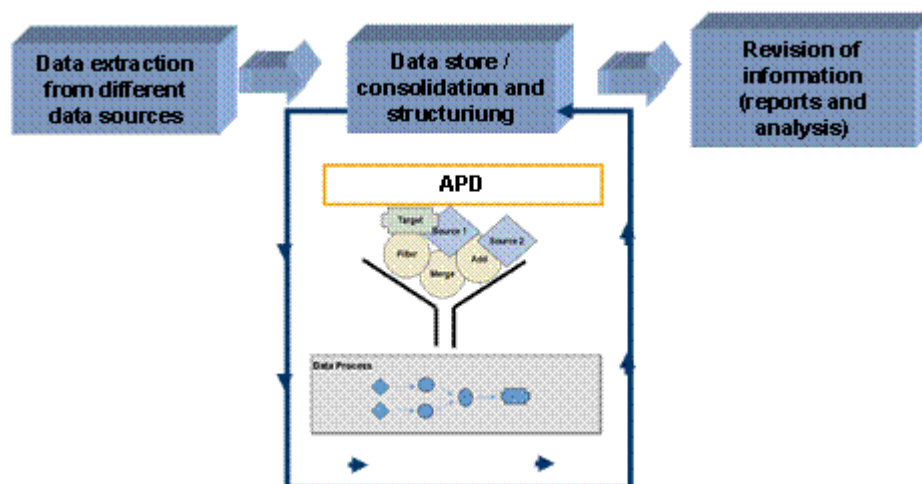
Examples of analysis processes include the calculation of ABC classes, determination of frequency distribution or of scoring information.

Integration

The analysis process designer is the application environment for the SAP data mining solution. The following data mining functions are integrated into the APD:

- Creating and changing data mining models
- Training data mining models with various BW data (data mining model as data target in the analysis process)
- Execution of data mining methods such as prediction with decision tree, with cluster model and integration of data mining models from third parties (data mining model as a transformation in the analysis process)
- Visualization of data mining models

The APD is integrated into the Administrator Workbench:



Restrictions

Integration into the Administrator Workbench has the following restrictions:

- The node texts are not language dependent.
- You can only integrate an analysis process into a process chain using the process type ABAP program.

To do this, choose the ABAP report `RSAN_PROCESS_EXECUTE`.

- Analysis processes are not displayed in the data flow display.
- The where-used list only functions from the analysis process to other objects and from data mining models to the analysis process, but not from other objects such as InfoProviders.

Functions

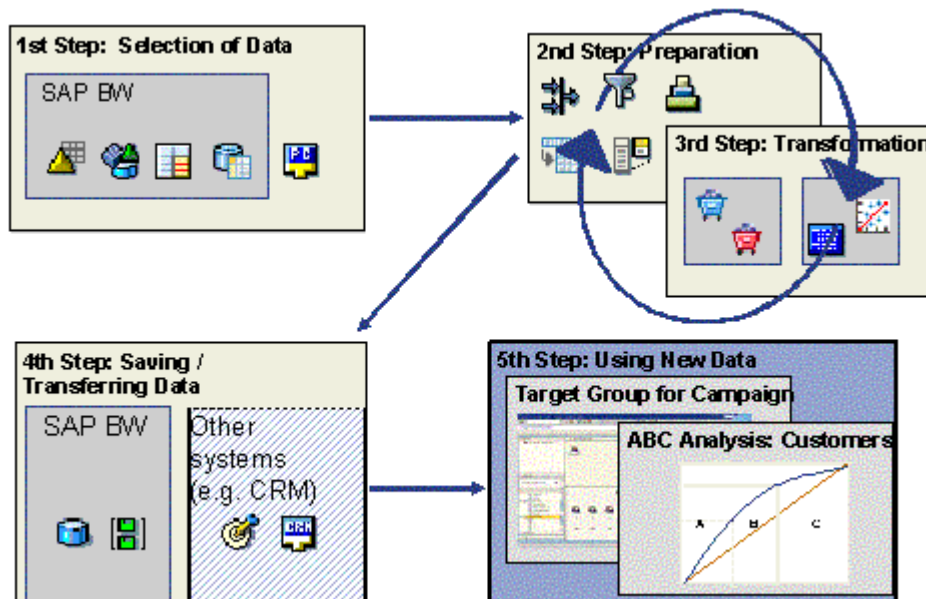
The analysis process designer is a workbench with an intuitive, graphic user interface for the creation, execution and monitoring of analysis processes. Analysis process can be created using Drag&Drop. Data from different data sources in the BW system can be combined, transformed and prepared for analysis in several individual steps so that it can then be resaved into targets in the BW system (transactional ODS object or InfoObjects with attributes) or in a CRM system. Various

- Data sources,
- Transformations and
- Data Targets

are available.

Various additional functions support you during modeling and executing an analysis process, as well as during interpretation of the analysis results.

The following graphic shows the various steps in the Analysis Process Designer.



First select a data target that contains the desired data. Then this data is prepared and then transformed. This transformed data is then saved in a BW object or in another system. For analysis, you can display the data in a query in the Business Explorer.

Versioning

Analysis processes are integrated into the versioning concept (active, inactive version, content version and delivery).

For more details on how the APD works with Direct-Update DSOs under different transformation, refer to my following whitepapers:

<http://www.sdn.sap.com/irj/scn/index?rid=/library/uuid/90e15fcc-b253-2e10-c4a6-e4593150f890>

<http://www.sdn.sap.com/irj/scn/index?rid=/library/uuid/d00dbb01-a753-2e10-a09b-81d9ad4b862b>

Creating DataStore Objects for Direct Update

When creating a DataStore object, you can change the DataStore object type under Settings via the context menu. The default setting is Standard. The switching of DataStore object types between standard and direct update is only possible if no data exists yet in the DataStore object.

Integration

Since DataStore objects for direct update cannot be filled with BI data using staging (data is not supplied from the DataSources), they are not displayed in the scheduler or in the monitor. However, you can update the data of DataStore objects of this type to additional InfoProviders.

If you switch a standard DataStore object, that already has update rules, to direct update, the update rules are set as inactive and can no longer be processed.

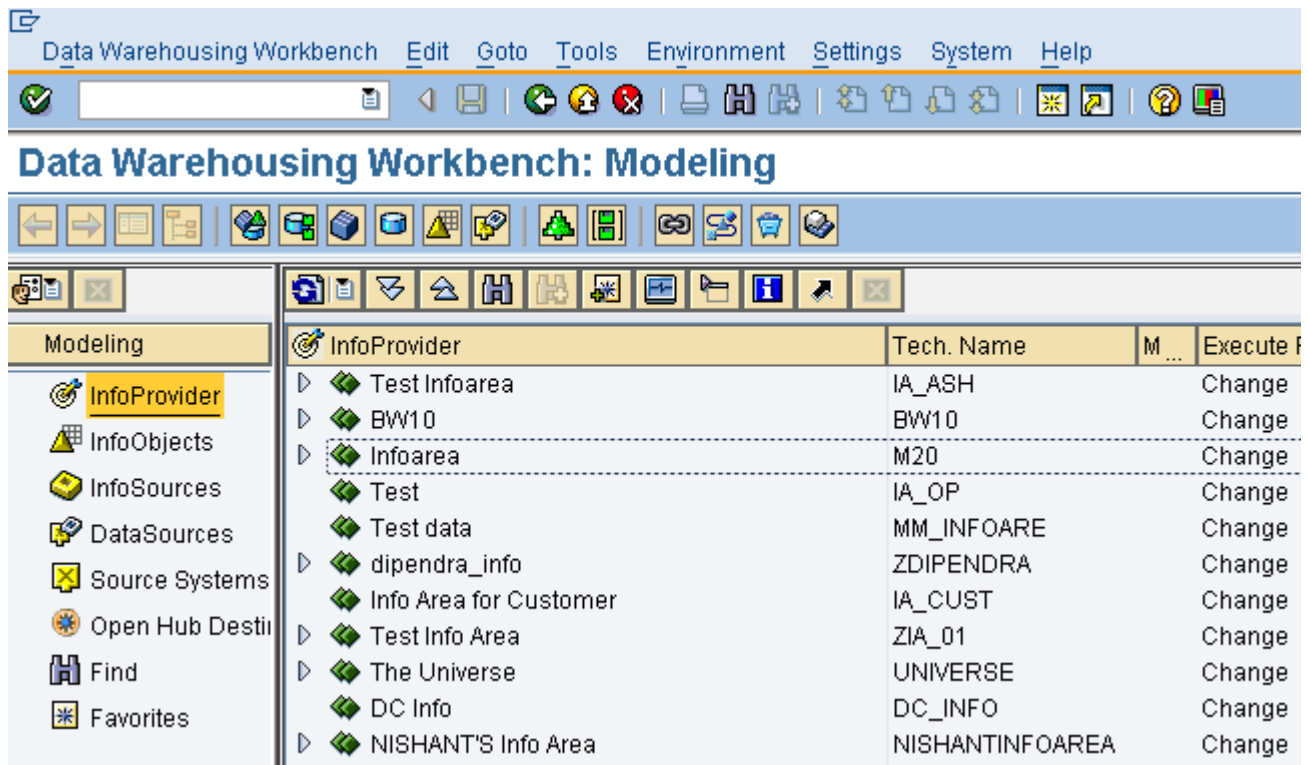
Since no change log is generated, delta update of InfoProviders stored at the end of the process is not possible.

The DataStore object for direct update is available as an InfoProvider in the BEx Query Designer and can be used in reporting.

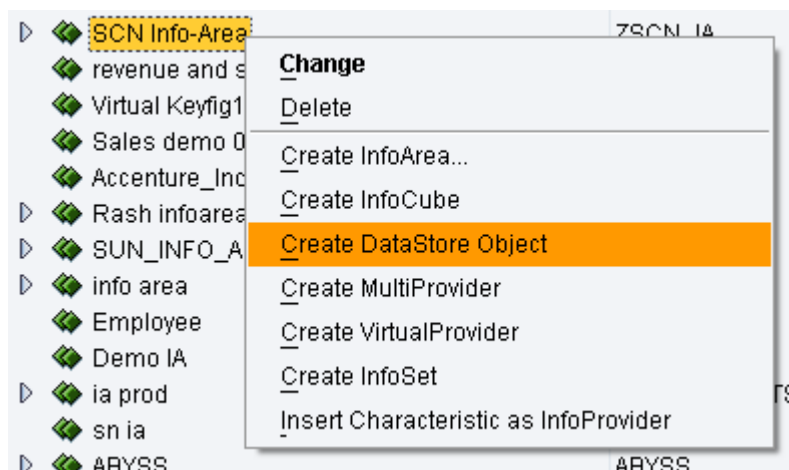
Designing a Direct-Update DSO

To create a Direct Update DSO, first go to **RSA1** transaction screen.

You will reach the DWW screen where you have to select **InfoProvider** under the **Modeling** pane present on the left side of the screen as shown below.



Now right click on your InfoArea and select **Create DataStore Object** from the context menu as shown below.



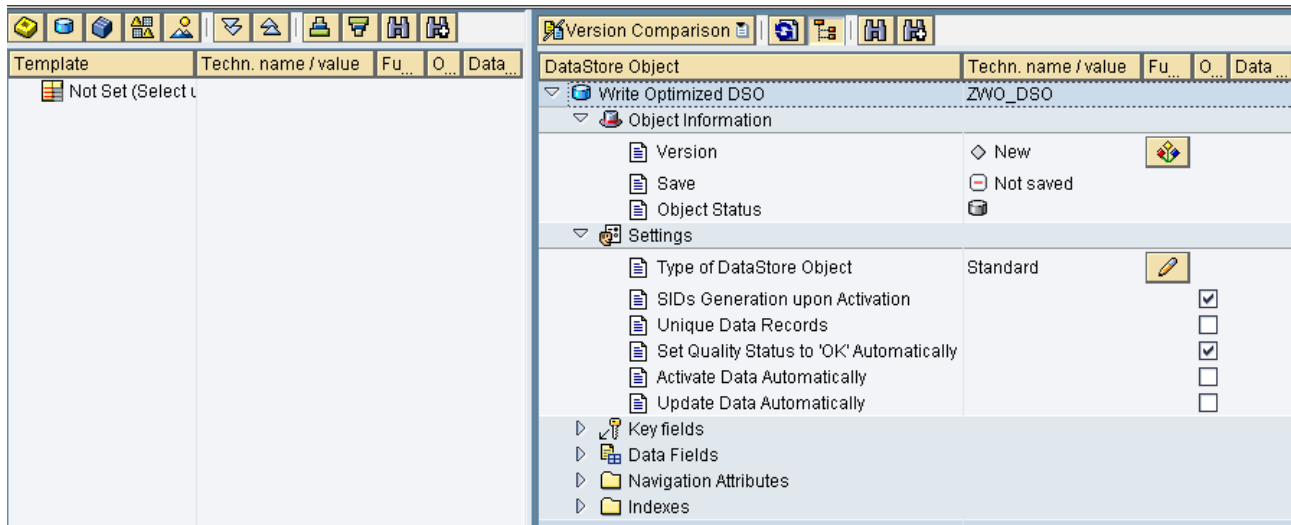
You will get the following window wherein you are required to fill in the technical name and description of the DSO to be created.

SAP has also provided an option of copying the entire DSO structure from another DSO using the **Copy from** Text-Field. This structure can be modified later.

We will use the Structure of the Standard DSO which we created in the first Part of the document for matters of simplicity and consistency.

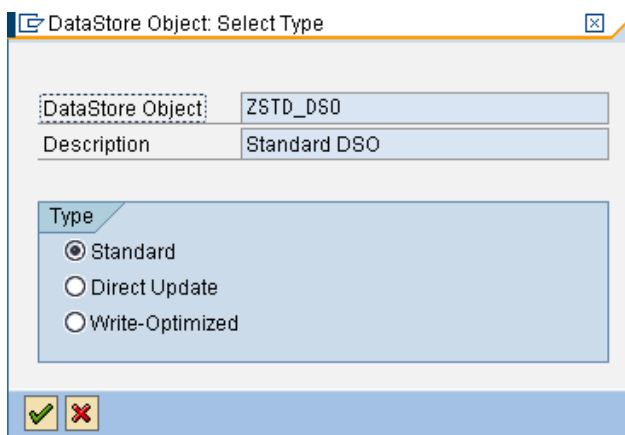
The filled in details are shown below. Press the Create  button to continue.

You will reach the following screen for editing the DSO.

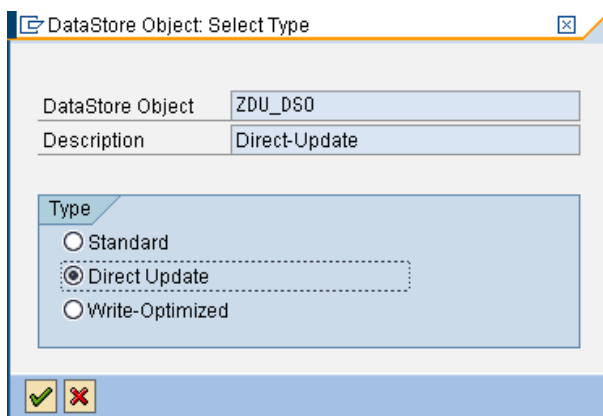


By Default, the DSO type is created as a standard type. This can be changed by clicking on the Change icon

You will get the following pop-up.



Now select the radio-button for Direct Update DSO, and go back to the previous screen by pressing



Now, the DSO has become a Direct Update one as seen below.

| DataStore Object | | Techn. name / value | Fu... | O... | Data ... | L | Key Fi... | C... | N... |
|--------------------------|--|---------------------|-------|------|----------|----|-----------|-------------------------------------|--------------------------|
| ▼ Direct-Update | | ZDU_DSO | | | | | | | |
| ▼ Object Information | | | | | | | | | |
| Version | | ◇ New | | | | | | | |
| Save | | ⊞ Not saved | | | | | | | |
| Object Status | | | | | | | | | |
| ▼ Settings | | | | | | | | | |
| Type of DataStore Object | | Direct Update | | | | | | | |
| ▼ Key fields | | | | | | | | | |
| PRODUCT | | ZPROD_SW | | | CHAR | 60 | | | |
| CUSTOMER | | ZCUS_SW | | | CHAR | 10 | | | |
| COST ELEMENT | | ZCEL_SW | | | CHAR | 10 | | | |
| cost character | | ZCCH_SW | | | CHAR | 20 | | | |
| Sales document | | ODOC_NUMBER | | | CHAR | 10 | | | |
| Calendar Year/Week | | OCALWEEK | | | NUMC | 06 | | | |
| ▼ Data Fields | | | | | | | | | |
| COST | | ZCOST_SW | | | CURR | 09 | Amount | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| REVENUE | | ZREV_SW | | | CURR | 09 | Amount | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| QUANTITY | | ZQUAN_SW | | | QUAN | 09 | Quantity | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Navigation Attributes | | | | | | | | | |
| Indexes | | | | | | | | | |

Settings in Direct Update DSO

As you can see, the following settings are available in a Direct Update DSO.

All of them will be explained one-by-one below in detail.

| | | | | | | | | | |
|--------------------------|--|---------------|--|--|--|--|--|--|--|
| ▼ Settings | | | | | | | | | |
| Type of DataStore Object | | Direct Update | | | | | | | |

Type of DataStore Object

This setting has been explained above while modifying the type of DSO from Standard to Direct Update.

Including Key Fields and Data Fields in the DSO

The DSO contains 2 kinds of fields namely, the key field and the data field.

The combination of key fields is responsible for uniquely identifying the data.

All other objects can be included as data fields.

All other objects can be included as data fields.

Similar to a Standard DSO, the same 2 ways are used to provide input InfoObjects into the DSO:


- Using Templates
- Using Direct Input


We have explained these methods in detail in the first part of the document.

Final Steps

Now our DSO structure design is complete.


Now we follow through the usual routine of Save, Check and Activate.

Save using the  button.


Press  to check for errors.

The following message confirms that there are no errors in design.



Press the  button to activate the DSO.

The Object Information menu now shows the DSO as active.

| DataStore Object | Techn. name / value | Fu... | O... |
|----------------------|----------------------|-------|---|
| ▼ Direct-Update | ZDU_DSO | | |
| ▼ Object Information | | | |
| Version | ◇ In Process | |  |
| Save | ⊕ Saved | | |
| Revised Version | = Active Version | | |
| Object Status | Ⓜ Active, executable | | |

The Direct Update DSO design is now complete.