

BW350

BI Data Acquisition

SAP NetWeaver

Date _____
Training Center _____
Instructors _____
Education Website _____

Participant Handbook

Course Version: 2006/Q2

Course Duration: 5 Days

Material Number: 50078949



An SAP course - use it to learn, reference it for work

Copyright

Copyright © 2006 SAP AG. 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 AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Trademarks

- Microsoft®, WINDOWS®, NT®, EXCEL®, Word®, PowerPoint® and SQL Server® are registered trademarks of Microsoft Corporation.
- IBM®, DB2®, OS/2®, DB2/6000®, Parallel Sysplex®, MVS/ESA®, RS/6000®, AIX®, S/390®, AS/400®, OS/390®, and OS/400® are registered trademarks of IBM Corporation.
- ORACLE® is a registered trademark of ORACLE Corporation.
- INFORMIX®-OnLine for SAP and INFORMIX® Dynamic ServerTM are registered trademarks of Informix Software Incorporated.
- UNIX®, X/Open®, OSF/1®, and Motif® are registered trademarks of the Open Group.
- Citrix®, the Citrix logo, ICA®, Program Neighborhood®, MetaFrame®, WinFrame®, VideoFrame®, MultiWin® and other Citrix product names referenced herein are trademarks of Citrix Systems, Inc.
- HTML, DHTML, XML, XHTML are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.
- JAVA® is a registered trademark of Sun Microsystems, Inc.
- JAVASCRIPT® is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.
- SAP, SAP Logo, R/2, RIVA, R/3, SAP ArchiveLink, SAP Business Workflow, WebFlow, SAP EarlyWatch, BAPI, SAPPHIRE, Management Cockpit, mySAP.com Logo and mySAP.com are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other products mentioned are trademarks or registered trademarks of their respective companies.

Disclaimer

THESE MATERIALS ARE PROVIDED BY SAP ON AN "AS IS" BASIS, AND SAP EXPRESSLY DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR APPLIED, INCLUDING WITHOUT LIMITATION WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WITH RESPECT TO THESE MATERIALS AND THE SERVICE, INFORMATION, TEXT, GRAPHICS, LINKS, OR ANY OTHER MATERIALS AND PRODUCTS CONTAINED HEREIN. IN NO EVENT SHALL SAP BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR PUNITIVE DAMAGES OF ANY KIND WHATSOEVER, INCLUDING WITHOUT LIMITATION LOST REVENUES OR LOST PROFITS, WHICH MAY RESULT FROM THE USE OF THESE MATERIALS OR INCLUDED SOFTWARE COMPONENTS.

About This Handbook

This handbook is intended to complement the instructor-led presentation of this course, and serve as a source of reference. It is not suitable for self-study.

Typographic Conventions

American English is the standard used in this handbook. The following typographic conventions are also used.

Type Style	Description
<i>Example text</i>	Words or characters that appear on the screen. These include field names, screen titles, pushbuttons as well as menu names, paths, and options. Also used for cross-references to other documentation both internal (in this documentation) and external (in other locations, such as SAPNet).
Example text	Emphasized words or phrases in body text, titles of graphics, and tables
EXAMPLE TEXT	Names of elements in the system. These include report names, program names, transaction codes, table names, and individual key words of a programming language, when surrounded by body text, for example SELECT and INCLUDE.
Example text	Screen output. This includes file and directory names and their paths, messages, names of variables and parameters, and passages of the source text of a program.
Example text	Exact user entry. These are words and characters that you enter in the system exactly as they appear in the documentation.
<Example text>	Variable user entry. Pointed brackets indicate that you replace these words and characters with appropriate entries.

Icons in Body Text

The following icons are used in this handbook.

Icon	Meaning
	For more information, tips, or background
	Note or further explanation of previous point
	Exception or caution
	Procedures
	Indicates that the item is displayed in the instructor's presentation.

Contents

Course Overview	vii
Course Goals	vii
Course Objectives	vii
Unit 1: Overview of BI.....	1
BI in SAP NetWeaver 2004s	3
Overview of Data Acquisition in BI.....	8
BI Content.....	41
Unit 2: Data Flow in Data Acquisition	53
Data Flow Overview in BI	54
Transformation Process.....	79
Direct Access to Source System Data	110
Real Time Data Acquisition.....	125
Unit 3: Data Acquisition with the Service API.....	143
Connecting SAP Source Systems to a BI System.....	145
Basics of the BI Service API.....	150
Transfer of Business Content DataSources	162
Logistics Data Extraction.....	166
Generic Data Acquisition.....	192
Enhancement of Business Content DataSources	219
Unit 4: Delta Management	237
Delta Management: Overview	239
Update Mode and Delta Process	245
Examples from Applications and Source Systems	277
Additional Features of Delta Data Acquisition.....	298
Unit 5: Transfer of Flat Files.....	303
Transfer of Flat Files.....	304
Unit 6: Data Acquisition with DB Connect	347
Data Acquisition with DB Connect.....	348
Unit 7: Data Acquisition with Universal Data Integration.....	369
Universal Data Integration	370

Unit 8: XML-Based Data Acquisition	391
Introduction to XML-Based Extraction.....	393
Using the Web AS SOAP Service (Optional).....	401
XML Data Acquisition Using a Web Service	408
XML Data Acquisition Using SAP PI	427
Unit 9: Data Acquisition with Third-Party ETL Tools	447
Data Transfer with Third-Party ETL Tools	448
Unit 10: Data Mart Interface	457
Data Mart Interface	458
Unit 11: Optional: Application-Specific Data Acquisition	487
Data Acquisition from Profitability Analysis (Optional)	488
Data Acquisition from Special Ledger (Optional)	512

Course Overview

Target Audience

This course is intended for the following audiences:

- Project team member with extensive knowledge of warehouse management and the Data Warehousing Administrator Workbench

Course Prerequisites

Required Knowledge

- Sound knowledge of the content that is covered in the BW310 – Data Warehousing course, especially working with the Data Warehousing Workbench

Recommended Knowledge

- Knowledge of the ABAP Dictionary and of at least one SAP component



Course Goals

This course will prepare you to:

- Attain knowledge and skills for data extraction from SAP components, and for the use of the required tools, such as DB Connect, SAP PI, UDI



Course Objectives

After completing this course, you will be able to:

- Describe the entire process of data extraction from SAP components
- Explain the data flow within BI

SAP Software Component Information

The information in this course pertains to the following SAP Software Components and releases:

Unit 1

Overview of BI

Unit Overview

The first lesson provides an overview of the SAP NetWeaver Business Intelligence, hereafter known as BI, technology platform. We will show you what position BI holds within SAP NetWeaver. In addition, key areas of BI will be described using the BI architecture in compressed form. From the perspective of this course, the second lesson will describe the architecture required for data staging with its components (source system types, DataSources, transformations, along with the various transfer mechanisms. In the last lesson, the BI Content is introduced with regard to its areas of application and components or objects. The procedure for transferring BI Content objects is also explained.



Unit Objectives

After completing this unit, you will be able to:

- Describe the concept behind SAP NetWeaver 2004s
- List the advantages and benefits of SAP NetWeaver 2004s
- Acquire an overview of the SAP NetWeaver 2004s architecture and its components
- Describe the position of BI in SAP NetWeaver 2004s
- Describe the BI architecture and its functional areas
- Name the different source systems and source system types
- Explain the importance of DataSources for data acquisition
- Describe the various transfer mechanisms
- Name BI Content (Business Content) objects and areas
- Explain the object versions
- Describe the transfer process for BI Content

Unit Contents

Lesson: BI in SAP NetWeaver 2004s	3
Lesson: Overview of Data Acquisition in BI	8
Procedure: Creating Data Transfer Processes Using Process Chains	21

Procedure: Creating Data Transfer Processes from the Object Tree in the Data Warehousing Workbench.....	24
Exercise 1: Overview of Data Transfer Process.....	27
Lesson: BI Content	41

Lesson: BI in SAP NetWeaver 2004s

Lesson Overview

This lesson provides you with an overview of the SAP NetWeaver 2004s technology platform. You will learn about the key areas of BI and the position of BI within SAP NetWeaver 2004s.



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the concept behind SAP NetWeaver 2004s
- List the advantages and benefits of SAP NetWeaver 2004s
- Acquire an overview of the SAP NetWeaver 2004s architecture and its components
- Describe the position of BI in SAP NetWeaver 2004s
- Describe the BI architecture and its functional areas

Business Example

Your company has recognized how useful it can be to integrate people, information and business processes in a heterogeneous system landscape and would like to benefit from this.

Practical experience has shown, however, that bringing together separate systems in individual projects by point-to-point integration is very costly and time-consuming, and makes such IT environments more and more inflexible.

To reduce complexity and costs, your company has decided to run the open and extensible SAP NetWeaver 2004s technology platform. This integration and application platform will enable your company to fully integrate all your employees, information and business processes across all systems, increase the return on investment (ROI), and reduce the total cost of ownership (TCO).

SAP NetWeaver 2004s

SAP NetWeaver 2004s is a comprehensive integration and application platform that supports you in bringing together all persons, information and business processes across all technological and enterprise boundaries. At the same time, SAP NetWeaver 2004s represents a web services-based platform that is available for all the solutions provided by SAP (such as SAP R/3 Enterprise, mySAP Business Suite, SAP xApps) and selected partners, as well as the basis for all future SAP and partner solutions.

Another outstanding feature of SAP NetWeaver 2004s is its openness and extensibility. SAP NetWeaver 2004s is compatible and can be enhanced with the standard market technologies - Microsoft .NET and IBM WebSphere - and supports the Java 2 Platform, Enterprise Edition (J2EE).

An additional advantage of SAP NetWeaver 2004s is the predefined and immediately deployable Business Content for individual components (such as Portal Content, BI Content), which reduces the time required for implementation.

SAP NetWeaver 2004s is the result of over 30 years of experience that SAP can demonstrate in enterprise solutions. Consequently, this technology guarantees a high degree of reliability, safety and scalability.

The advantages of SAP NetWeaver 2004s as described above lead on the one hand to increased ROI, as SAP NetWeaver 2004s protects your existing IT investments by integrating them, and incorporating systems profitably that are already used in your company. On the other hand, it reduces the TCO, meaning the total cost of IT investment over the course of its life cycle. As well as investment costs, this includes costs for maintenance, administration, support, integration, and so on.

The following graphic shows the main benefits of SAP NetWeaver 2004s.



SAP NetWeaver is an integration and application platform ...

- that brings together people, information and business processes across all technological and organizational boundaries
- that enables a simplified and flexible IT infrastructure
- that is based on Web Service
- that provides openness and extensibility
- that is extremely reliable, safe and scalable
- that provides a platform for all SAP and partner solutions
- that provides predefined Business Content for the individual components (such as Portal Content, BI Content and so on)
- which maximizes the return on investment (ROI)
- that minimizes the total cost of ownership (TCO)

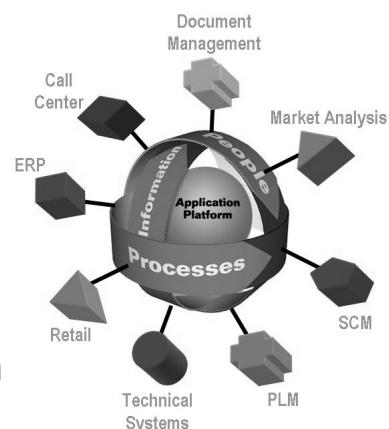


Figure 1: Benefits of SAP NetWeaver 2004s

BI in SAP NetWeaver 2004s

BI is the Business Intelligence component for SAP NetWeaver 2004s. The following graphic shows where BI is positioned within SAP NetWeaver 2004s. It also lists the functional areas of BI. These will be explained in greater detail in the text that follows.

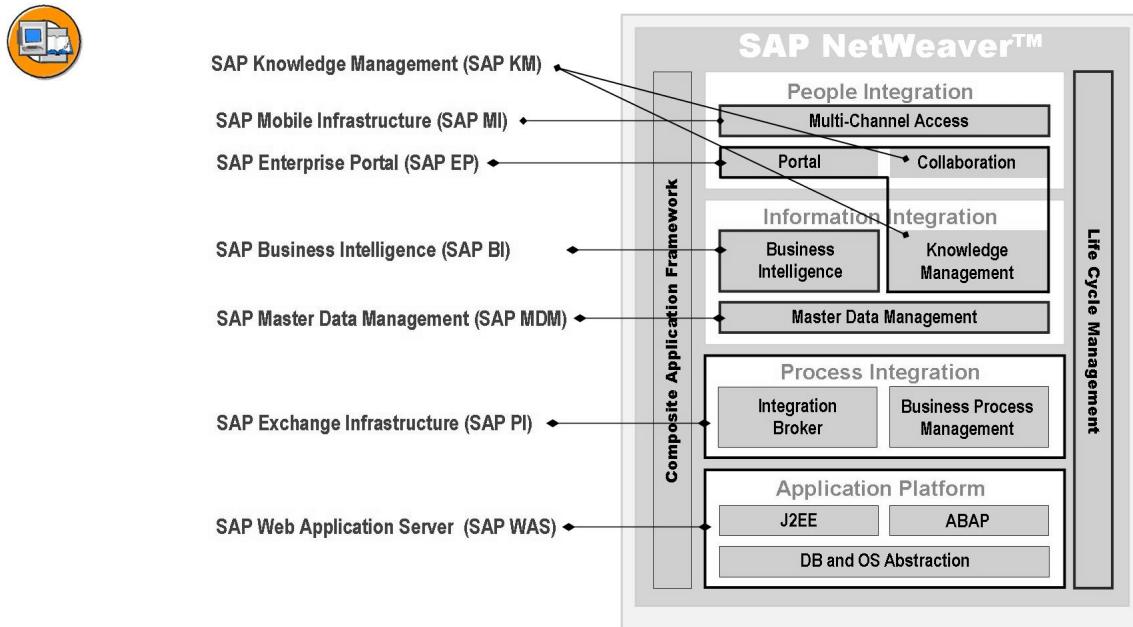


Figure 2: SAP NetWeaver 2004s: Integration and Application Platform

Data warehousing in BI means integrating, transforming, consolidating, cleaning up and storing data as well as staging data, for example for reporting, or for non-SAP data marts. The data warehousing process includes modeling data, staging data and administration. The central tool for data warehousing tasks in BI is the Administrator Workbench.

The **BI Platform** (Business Intelligence Platform) provides the technological infrastructure as well as various analytical technologies and functions. This includes the OLAP Processor, the Meta Data Repository, BPS (Business Planning and Simulation) as well as the Analysis Process Designer and data mining.

The **Business Explorer** (BEx) is the presentation suite for BI. It provides you with flexible reporting and analysis tools for strategic analysis and for operative reporting in your company. These tools include query, reporting and analysis functions. You can evaluate historical and current data in various degrees of detail and from different perspectives, both over the Web and in MS Excel. With the help of BEx Information Broadcasting you can distribute BI content from BI by e-mail as prepared documents with historical data or as links with live data, or

publish them using SAP EP functions. BEx provides a broad spectrum of users with access to information in BI, such as information about SAP EP, SAP MI and the Web Application Designer.

The functional areas above are presented in the following section on BI architecture:

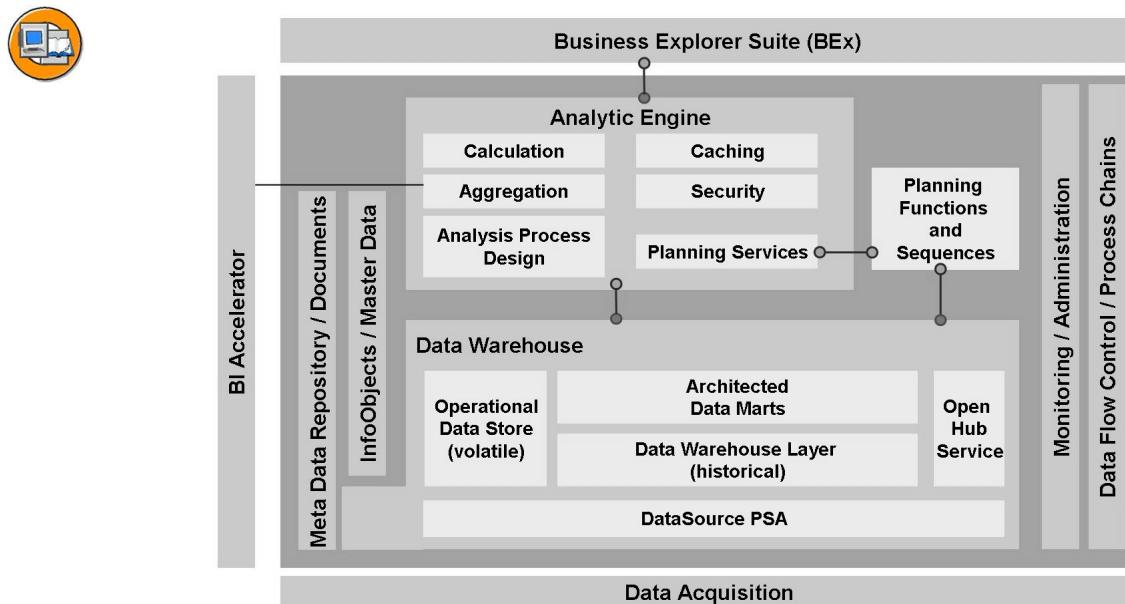


Figure 3: BI: Architecture

BI offers a broad, open, standards-based architecture. You can extract data from different systems to an BI system and evaluate this data for reporting with different front-end tools, or distribute it to other systems. The transfer mechanisms (XML/SOAP, BAPI, DB Connect, File, Open Hub Service and Universal Data Integration) presented in the graphic for data staging are the focal point of this course, and are dealt with in detail in the following units.



Lesson Summary

You should now be able to:

- Describe the concept behind SAP NetWeaver 2004s
- List the advantages and benefits of SAP NetWeaver 2004s
- Acquire an overview of the SAP NetWeaver 2004s architecture and its components
- Describe the position of BI in SAP NetWeaver 2004s
- Describe the BI architecture and its functional areas

Lesson: Overview of Data Acquisition in BI

Lesson Overview

This lesson provides you with an overview of the various source system types and the different transfer mechanisms for data acquisition. The DataSource together with the different options for data transformation in the data acquisition process, are also explained.



Lesson Objectives

After completing this lesson, you will be able to:

- Name the different source systems and source system types
- Explain the importance of DataSources for data acquisition
- Describe the various transfer mechanisms

Business Example

For reporting in SAP NetWeaver Business Intelligence, your company needs data from diverse data sources, such as SAP systems, non-SAP systems, the Internet and other business applications. You should therefore examine the technologies that SAP NetWeaver Business Intelligence offers for data acquisition.

Data Acquisition in BI

Data acquisition is one of the data warehouse processes in SAP NetWeaver Business Intelligence (BI). BI provides mechanisms for staging data (master data, transaction data, metadata) from various sources. You can determine whether BI is the target or the source of the data transfer:

If data is staged from different sources for transfer to an BI system, BI is the target system for the data transfer. If, on the other hand, data in BI is staged for distribution within BI, or for distribution to analytical and other applications, BI is then the source system for the data transfer.

The graphic below shows a simplified version of the data acquisition architecture. The following section deals with the architecture components below:

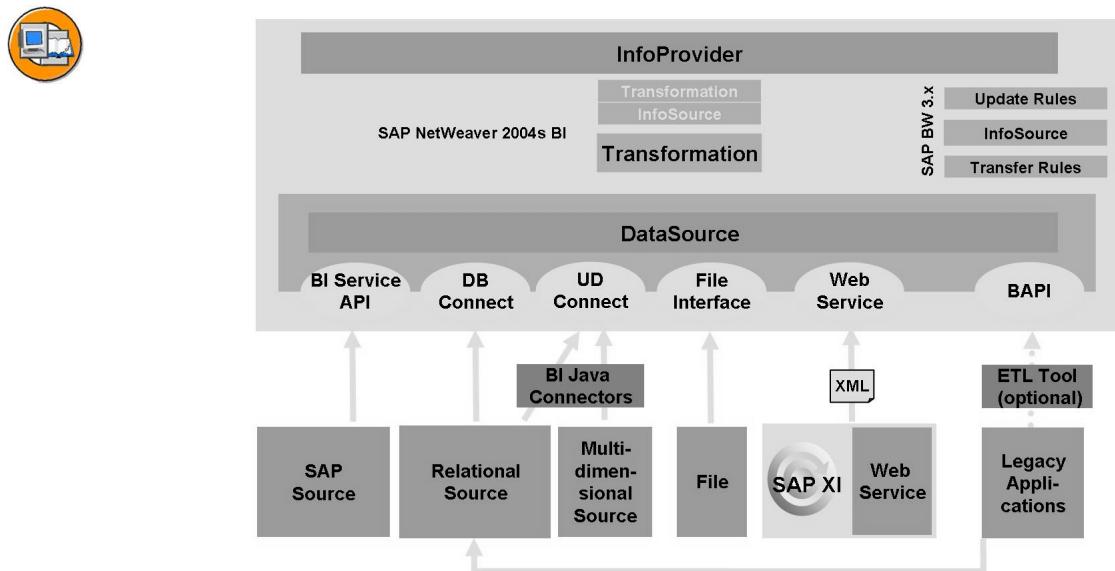


Figure 4: BI Data Acquisition: Architecture

The graphic below shows a simplified version of the data acquisition data flow within BI.

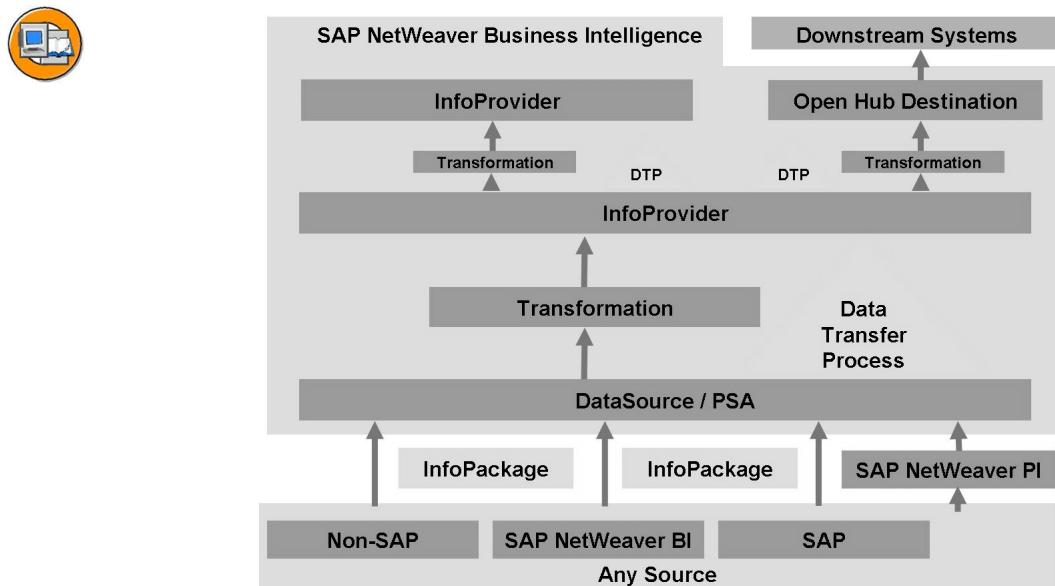


Figure 5: BI Data Acquisition Data Flow

Source Systems and Source System Types

Source systems are all systems that stage data for BI. A distinction is made between the following source system types:

1. SAP systems
2. BI systems
3. Flat files for which metadata is maintained manually and transferred to BI using a file interface
4. Database management systems into which data is loaded from a database supported by SAP using DB Connect, without using an external extraction program
5. Relational or multidimensional sources that are connected to BI using UD Connect
6. Web Services that transfer data to BI by means of a push
7. Non-SAP systems for which data and metadata is transferred using staging BAPIs

The source system type is determined in the source systems area of the Data Warehousing Workbench, as shown in the following graphic:

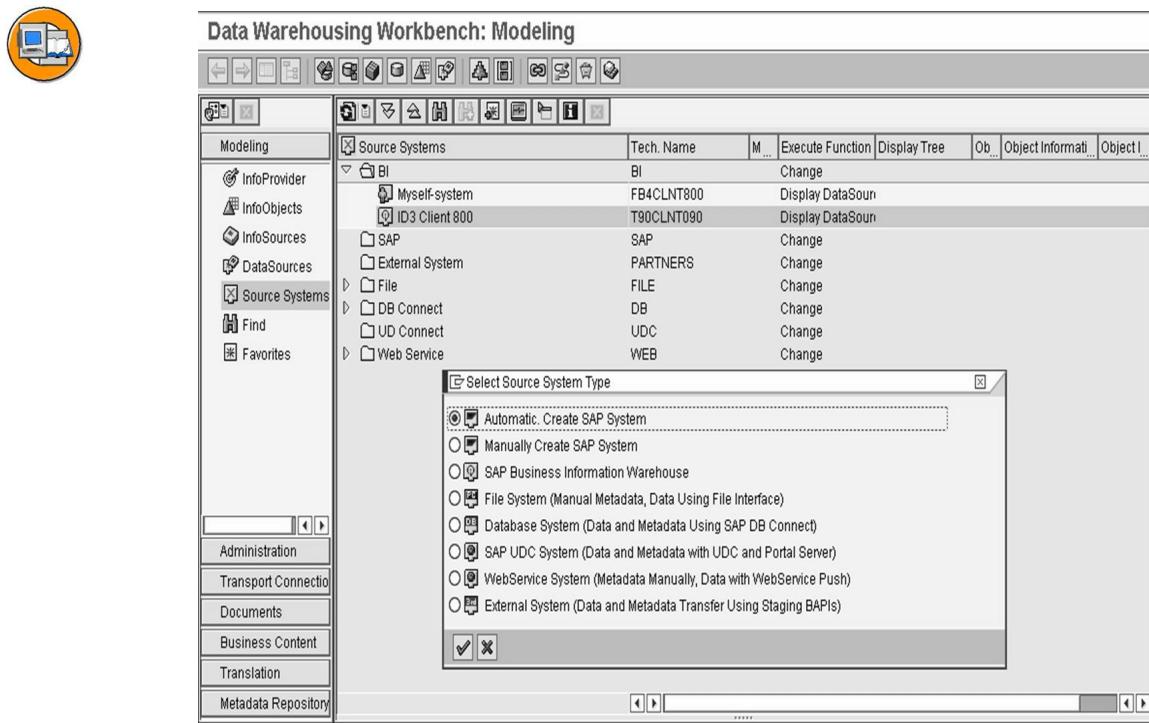


Figure 6: Setting Up Source Systems in BI



Hint:

1. Setting up the individual source system types varies a great deal, and will therefore be explained in the following unit with reference to each source system type.
2. If the source system type is an BI system, it is referred to as an BI data mart (refer to the unit on the data mart interface).
3. By default, each BI system has a link to itself. This source system type is created by default, which means that you do not need to make any more settings here. It is used for data transfer scenarios between distinct BI systems. Transferring data between InfoProviders within a single BI system uses a Data Transfer Process (DTP).

Data Acquisition in BI

DataSource

A DataSource comprises the amount of fields that provide the data for a business unit for data transfer to BI. From a technical viewpoint, the DataSource includes a quantity of fields that logically belong together that are offered for data transfer into BI in a flat structure (the extraction structure), or for hierarchies they are offered in multiple flat structures.

A distinction is made between the following DataSource types:

- *DataSources for transaction data*
- *DataSources for master data*
- - *DataSources for attributes*
 - *DataSources for texts*
 - *DataSources for hierarchies*



Hint: SAP delivers predefined DataSources for SAP source systems. .

From the perspective of the source system, a DataSource contains the metadata that describes the data to be extracted and how this data is to be extracted. These are specific to each source system type and each DataSource within the source system.

From the perspective of the particular source system, the metadata for a DataSource includes:

- A number of logically related fields that are offered in a flat structure, the extraction structure, for transferring data to BI.
- *Application component*
- *Extraction method*
- *Extractor*
- *Delta process*
- *Transfer method*
- ...

The data is loaded from any source in the DataSource structure into BI, using an InfoPackage. You determine the target into which the data from the DataSource is to be updated in the transformation. You also assign fields for the DataSource to InfoObjects from the target object in BI.

InfoSource

An InfoSource is a non-persistent structure consisting of InfoObjects for joining two transformations.

Normally a transformation directly links from a source DataSource (or InfoProvider) to a target InfoProvider and an InfoSource is not needed. You use InfoSources if you want to run two sequential transformations in the data flow without storing the data again. If you do not have transformations that run sequentially, you can model the data flow without InfoSources.

In contrast to the InfoSource 3.x, as of Release SAP NetWeaver BI 2004s, an InfoSource behaves like an InfoSource 3.x with flexible update. The data in an InfoSource is updated to an InfoProvider using a transformation. You can identify InfoObjects in the InfoSource as keys. In the transformation, the data records are then aggregated using this key.

The following graphic shows how InfoSources are integrated in the dataflow:

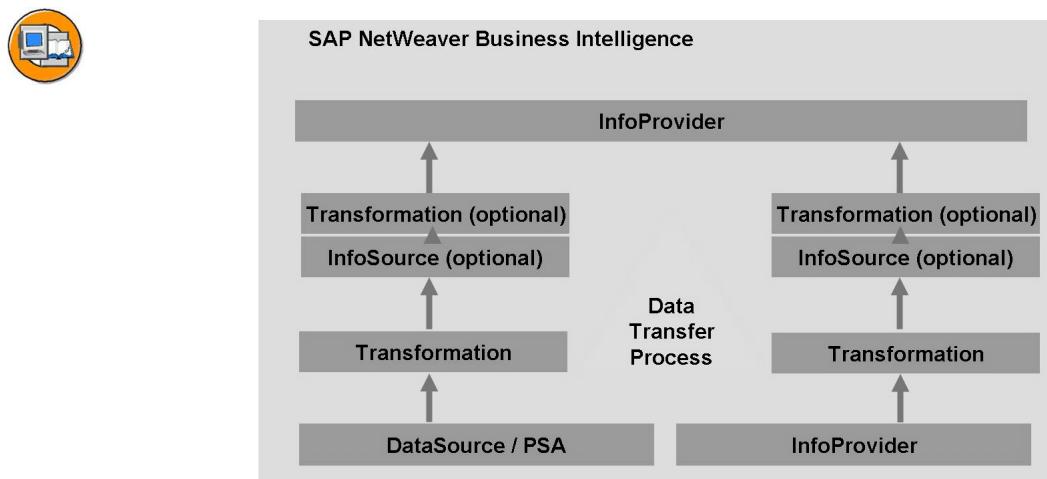


Figure 7: Enhanced Data flow in SAP NetWeaver 2004s BI

The InfoPackage controls the transfer of data from the source to the entry layer of BI. The data transfer process controls the distribution of data within BI. The graphic illustrates an example of a data update from the DataSource to an InfoProvider. The data can be updated from an InfoProvider to another InfoProvider using a data transfer process. This can be seen from the following graphic:

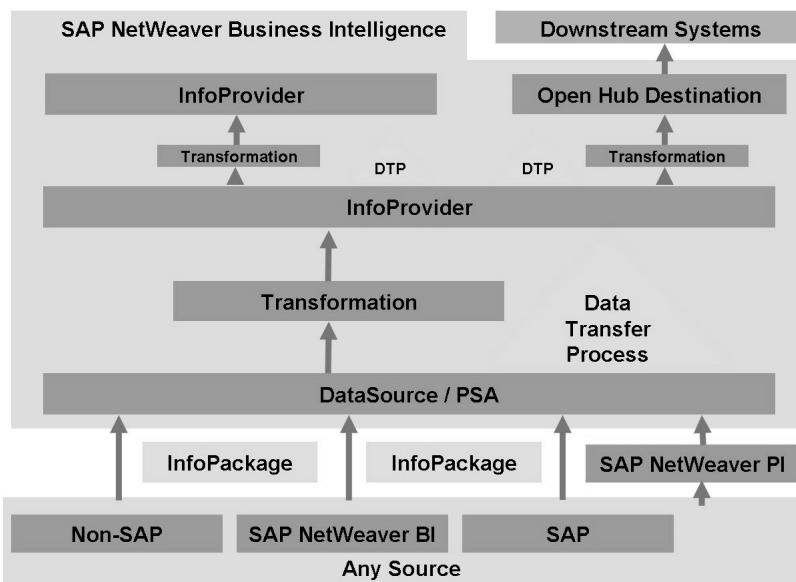


Figure 8: InfoPackages in SAP NetWeaver 2004s BI Data Flow

The data transfer process can also be used to control data distribution from a BI system into any target outside of the BI system. For this purpose, a data transfer process with an open hub destination is used as the target.

Data Transformation Process

The transformation process allows you to define rules and consolidate, cleanse, and integrate data. You can define semantic keys for the aggregation. In the SAP NetWeaver 2004s transformation concept you no longer require two different rules, a transfer rule and an update rule; you only need one transformation rule. You edit the transformation rule on an intuitive graphic user interface. InfoSources are no longer mandatory; they are optional and are only required for certain functions. Transformations also provide additional functions such as quantity conversion and the option to create an end routine or expert routine.

In releases before SAP NetWeaver 2004s, the central object for the transformation is the InfoSource 3.x. The individual fields of the DataSource 3.x are assigned to the relevant InfoObjects in it. The data can then be transformed using transfer rules. Update rules 3.x specify how the data (characteristics, key figures, time characteristics) is updated into the InfoProvider from the communication structure of an InfoSource 3.x. The data can also be transformed in the update rules 3.x.

You use the data transfer process (DTP) to transfer data within BI from a persistent object to another object in accordance with certain transformations and filters. In this respect, it replaces the data mart interface and the InfoPackage. As of SAP NetWeaver 2004s, the InfoPackage only loads data to the entry layer of BI (PSA). 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. For example, you can use filters between a DataStore object and an InfoCube. Data transfer processes are used for standard data transfer, for real-time data acquisition, and for accessing data directly.

The following graphic illustrates the transformation process.

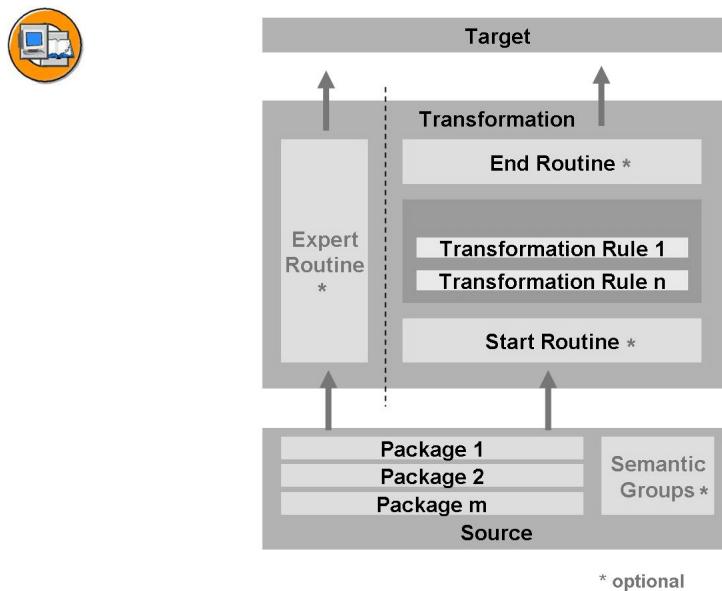


Figure 9: Transformation in SAP NetWeaver 2004s BI

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 extraction 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.

The data transfer process supports you in handling data records with errors. The data transfer process also supports error handling for DataStore objects. When you define the data transfer process, you can determine how the system responds to errors. At runtime, the incorrect data records are sorted and written to an error stack (request-based database table). A special error DTP further updates the data records from the error stack into the target. It is easier to restart failed load

processes if the data is written to a temporary storage after each processing step. It also allows you to find records that have errors. In the extraction monitor for the data transfer process request or in the temporary storage for the processing step (if filled), you can display the data records in the error stack . In data transfer process maintenance, you determine the processing steps after which you want to store data temporarily.

It is recommended to use process chains. You use a process chain to define a data transfer process. Alternatively, you can define a data transfer process for an InfoProvider in an object tree in the Data Warehousing Workbench. 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.

Transformations also provide additional functions such as quantity conversion and the option to create an end routine or expert routine. The Data Transformation Process will be discussed in the lesson on 'Transformation Features.'

Handling Data Records with Errors

For a data transfer process (DTP), you can specify how the system reacts to incorrect data records. If you activate error handling, the records with errors are written to a request-based database table, the error stack. You can use a special data transfer process, the error DTP, to post the records to the target.

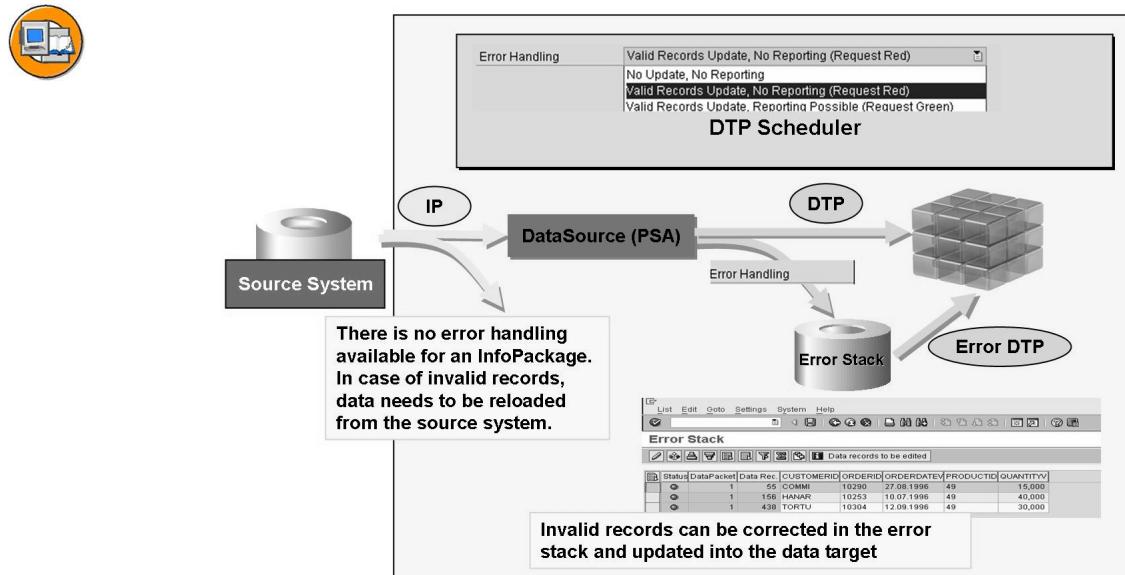


Figure 10: DTP Error Handling Overview

You can create an error DTP for an activated data transfer process on the *Update* tab page, and 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 the data from the error stack (in this case, the source of the DTP) and transfer it to the target that you have already defined in the data transfer process. The temporary storage at the processing step level of the DTP request allows you to determine the step in which the error occurred.

1. Define on the *Update* tab page how the system reacts to data records with errors:

- No update, no reporting (default)

If errors do occur, the update of the entire data package is canceled. The request is not released for reporting. However, the system continues to check the records

- Updating valid records, no reporting (request red)

This option enables you to update valid data that is released for reporting only after the administrator has checked the incorrect, not updated records, and has manually released the request (by a QM action, meaning setting the Overall status on the tabstrip *Status* in the extraction monitor).

- Updating valid records, reporting possible

The valid records can be reported on immediately. Automatic follow-up actions are also carried out, such as adjusting the aggregates.

- Specify the maximum number of incorrect data records allowed before the transfer process is terminated. If you do not make an entry here, the treatment of the incorrect data records is not activated, and the update is terminated with the first error.
- Select how the system should react when the number of data records received is not the same as the number of data records updated.

A difference between the number of records received and the number of updated records can occur if the records are sorted, aggregated, or added in the transformation rules or during the update.

If you set the *No Aggregation* indicator, the request is taken as incorrect if the number of received records is not the same as the number of updated records.

Independently of this indicator, an error arises when the number of selected records is not the same as the number of records received.

2. On the *Update* tab page, define the key fields for the error stack. This is known as a Semantic Group. The key should be as detailed as possible (maximum of 16 key fields). The fewer the number of key fields defined, the more records are updated to the error stack. In the standard setting the key for the DataStore object is defined as the key for the error stack.
3. Make the settings for the temporary storage by choosing *Goto → Settings for DTP Temporary Storage*. Here you can define the DTP request processing steps required for temporary storage (for example, extraction, filtering, removing new records with the same key, and transformation), as well as the level of detail the data is temporarily stored with, and when the temporary storage is to be deleted again.
4. Once the data transfer process has been activated, create an error DTP on the *Update* tab page, which you then include in a process chain or (in the case of an error) start manually to post the data with errors to the target

Error Stack

An Error Stack is a request-based table into which the incorrect data records from a data transfer process are written.

At run time, the incorrect data records are written into an error stack if the error handling for the data transfer process has been activated. You use the error stack to post the data to the target destination once the error has been removed.



DTP Error Stack - Summary

	No Error handling (InfoPackage)	No update, no reporting	Update valid records, no reporting	Update valid records, reporting possible
Monitor entry	X	X	X	X
Abort of update	X	X		
Upd. valid records			X	X
Marked in tem. storage		X	X	X
Update into Error Stack			X	X
Color of Request	red	red	red	green

Figure 11: DTP Error Stack Summary

In the extraction monitor for the data transfer process you can branch to PSA maintenance by choosing *Error Stack* in the toolbar and display and edit the records with errors in the error stack. With an error DTP, you can post the data records manually or by means of a process chain to the target. Once the data records have been successfully posted, they are deleted from the error stack.

- **Note:** If a request is deleted in a DataStore object, the associated data records are also deleted in the error stack.

Temporary Storage

Temporary storage is a table that contains the data records processed with a request in a step together with a status display.

The temporary storage is used at run time to allow processes to be restarted when errors occur without a significant impact on performance. It also allows you to find records with errors.

You can display the temporary storage for the different processing steps using the extraction 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.

- **Note:** In data transfer process maintenance you can determine when the temporary storage is to be deleted. This can happen either after the request has been posted successfully to the target, when the request is deleted, or after a certain interval (configured by you) has passed since the request was processed.

Extraction Monitor

The request status and request messages from the data transfer processes (DTPs) and the InfoPackage load processes are integrated in the extraction monitor that you can call in the Data Warehousing Workbench in the function area *Administration*, under *Monitors*.

You can also call the extraction monitor using data transfer process maintenance, using the process monitor on a DTP in the log view of the process chain maintenance, or using the *Call Monitor* function for a target object in an object tree in the Data Warehousing Workbench.

The extraction monitor for a data transfer process request shows you detailed information about processing the request, in addition to the information about the header.

At the first level of the detail tree, the system displays the status of the requests at various points during the process, for example, at generation, when processing the steps during the DTP program, and when setting the technical and complete status of the request. The status display for processing is then divided by the various steps of the data transfer process and displays the status and messages for the individual processing steps and data packages in the processing steps.

At the level of the processing step, you can also branch into the temporary storage display for this particular processing step (if you have defined in DTP maintenance that the temporary storage is to be filled after this processing step). You can branch to PSA maintenance for the error stack if the data was written to the error stack and has not been updated yet. You can also branch to the administration for the source and the target (DataSource or InfoProvider), as well as to the job and process overviews.



Creating Data Transfer Processes Using Process Chains

Use

You use the data transfer process (DTP) to transfer data from source objects to target objects within BI. You can also use the data transfer process to access InfoProvider data directly.

Procedure

1. You are in the plan view of the process chain in which you want to include a data transfer process as a process. The process type *Data Transfer Process* is available in the *Load Process and Postprocessing* process category.
2. Use Drag & Drop or double-click to include the process in the process chain.
3. To create a data transfer process as a new process variant, enter a technical name and choose *Create*.
The dialog box for creating a data transfer process appears.
4. Select the type of data transfer process:
 - For VirtualProviders, the only available option is *DTP for Direct Access*.
 - For DataStore objects, choose *Standard DTP* or *DTP for Real-Time Data Acquisition*.
5. Select the source object type and the object from which you want to transfer data into the target. When you select the source or target object, input help is available. Input help shows you the selection of objects that already exist in the data flow for the source or target object. If only one object exists in the data flow, this is selected by default. An additional List pushbutton is available. This allows you to select an object from the complete list of BI objects.
6. Choose *Continue*.

The data transfer process maintenance screen appears. The header data for the data transfer process shows the description, ID, version, and status of the data transfer process, along with the delta status

Continued on next page

7. On the *Extraction* tab page, determine the parameters
 - Choose the extraction mode. You can choose full mode or delta mode. You do not need to initialize the delta process explicitly for the delta transfer
 - If necessary, *determine filter criteria* for the delta transfer. This means that you can use multiple data transfer processes with disjunctive selection conditions to transfer small sets of data from a source efficiently into one or more targets, instead of transferring large volumes of data. You can specify individual selections, multiple selections, intervals, selections based on variables, or routines. Choose *Change Selection* to change the list of InfoObjects that can be selected. Apply further settings, which are dependent on the source object.
 - The icon next to pushbutton Filter indicates that predefined selections exist for the data transfer process. The tooltip for this icon displays the selections as a character string.
 - Choose **Semantic Groups** to specify how you want to build the data packages that are read from the source. To do this, you determine key fields. Data records that have the same key are combined in one data package. Currently, only data that has been read from the PSA can be processed further in semantic groups. This setting also determines the key fields for the error stack.
8. On the *Update* tab page, determine the parameters. Apply settings for error handling: Determine how the system updates the valid records when errors occur, the number of errors allowed before the load process terminates, whether the load process should be judged to contain errors if records are being aggregated, sorted, or added during the transformation
Under *Semantic Groups* determine the key fields for the error stack.
9. Make the settings for the temporary storage by choosing *Goto → Settings for DTP Temporary Storage* On this tab page, the process flow of the program for the data transfer process is displayed in a tree structure. Specify the status that you want the system to adopt for the request if warnings are displayed in the log. Specify how you want the system to determine the overall status of the request.
The system automatically determines the processing mode for the background processing of the respective data transfer process.
10. Check the data transfer process, save and activate it.

Result

When you go back to process chain maintenance, the data transfer process of the plan view is displayed and you can include it in your process chain. When you activate and schedule the chain, the system executes the data transfer process as

Continued on next page

soon as it is triggered by an event in the predecessor process in the chain. If you do not include the data transfer process in a process chain, you can execute it in the background.

If you want to execute a standard data transfer process in debug mode, choose *Serially in Dialog Process (for Debugging)* as the processing mode on the Execute tab page. In this case, you can determine breakpoints in the overview for the process flow of the program.



Creating Data Transfer Processes from the Object Tree in the Data Warehousing Workbench

1. You are in the plan view of the process chain in which you want to include a data transfer process as a process. The process type *Data Transfer Process* is available in the *Load Process and Postprocessing* process category.
2. Use Drag & Drop or double-click to include the process in the process chain.
3. Select the type of data transfer process:
 - For VirtualProviders, the only available option is *DTP for Direct Access*.
 - For DataStore objects, choose *Standard DTP* or *DTP for Real-Time Data Acquisition*.
4. Select the source object type and the object from which you want to transfer data into the target. When you select the source or target object, input help is available. Input help shows you the selection of objects that already exist in the data flow for the source or target object. If only one object exists in the data flow, this is selected by default. An additional List pushbutton is available. This allows you to select an object from the complete list of BI objects.
5. Choose *Continue*.

The data transfer process maintenance screen appears. The header data for the data transfer process shows the description, ID, version, and status of the data transfer process, along with the delta status

Continued on next page

6. On the *Extraction* tab page, determine the parameters
 - Choose the extraction mode. You can choose full mode or delta mode. You do not need to initialize the delta process explicitly for the delta transfer
 - If necessary, *determine filter criteria* for the delta transfer. This means that you can use multiple data transfer processes with disjunctive selection conditions to transfer small sets of data from a source efficiently into one or more targets, instead of transferring large volumes of data. You can specify individual selections, multiple selections, intervals, selections based on variables, or routines. Choose *Change Selection* to change the list of InfoObjects that can be selected. Apply further settings, which are dependent on the source object.
 - The icon next to pushbutton Filter indicates that predefined selections exist for the data transfer process. The tooltip for this icon displays the selections as a character string.
 - Choose **Semantic Groups** to specify how you want to build the data packages that are read from the source. To do this, you determine key fields. Data records that have the same key are combined in one data package. Currently, only data that has been read from the PSA can be processed further in semantic groups. This setting also determines the key fields for the error stack.
7. On the *Update* tab page, determine the parameters. Apply settings for error handling: Determine how the system updates the valid records when errors occur, the number of errors allowed before the load process terminates, whether the load process should be judged to contain errors if records are being aggregated, sorted, or added during the transformation
Under *Semantic Groups* determine the key fields for the error stack.
8. Make the settings for the temporary storage by choosing *Goto → Settings for DTP Temporary Storage* On this tab page, the process flow of the program for the data transfer process is displayed in a tree structure. Specify the status that you want the system to adopt for the request if warnings are displayed in the log. Specify how you want the system to determine the overall status of the request.
The system automatically determines the processing mode for the background processing of the respective data transfer process.
9. Check the data transfer process, save and activate it.

Exercise 1: Overview of Data Transfer Process

Exercise Objectives

After completing this exercise, you will be able to:

- Create a generic DataSource
- Use a migrated DataSource to load data into an InfoCube
- Use a Process Chain to load data into an InfoCube

Business Example

Before you begin exploring the data acquisition process in BI, you need to review the basic principles by building a generic DataSource and using it to load data into an InfoCube.

Task 1:

Create an InfoCube and give it the name and description **B350DS##**.

1. Create the InfoCube **B350DS##** as a copy.

Field Name	Values
<i>InfoCube</i>	B350DS##
<i>Long Description</i>	GR## DTP Overview
<i>Copy From</i>	B350DS00

Task 2:

Create a generic DataSource for the source system **T90CLNT090** using the table **SFLIGHT** as a basis.

1. Create a generic DataSource **Z_SFLIGHT_##** of type “Transaction Data” for the source system **T90CLNT090**.

<i>Appl. Component</i>	BW350-##
<i>Short description</i>	GR## Flight DS
<i>Medium description</i>	GR## Flight DataSource
<i>Long description</i>	GR## Flight DataSource
<i>View/Table</i>	SFLIGHT

Continued on next page

Task 3:

You now need to replicate your new DataSource to BI.

1. Replicate your DataSource as a 3.x DataSource in BI system.

Task 4:

You discovered that you made a mistake by creating a 3.x DataSource, so you will have to migrate the 3.x DataSource to a BI DataSource.

1. Migrate your 3.x Datasource **Z_SFLIGHT_##** to a BI DataSource.

Task 5:

Create a Transformation between your Data Source **Z_SFLIGHT_##** and your InfoCube **B350DS##**.

1. From the context menu of your DataSource create a Transformation between your DataSource and your InfoCube

<i>Object Type</i>	InfoCube
<i>Name</i>	B350DS##

Use the following fields:

<i>CARRID</i>	CARR_ID
<i>CONNID</i>	CONN_ID
<i>FLDATE</i>	OCALDAY
<i>PRICE</i>	PRICE
<i>CURRENCY</i>	PRICE
<i>PLANETYPE</i>	PLANETYPE
<i>SEATSMAX</i>	SEATSMAX
<i>SEATSOCC</i>	SEATOCC

Task 6:

Create a Data Transfer Process between your InfoCube **B350DS##** and your DataSource **Z_SFLIGHT_##**.

1. Create a Data Transfer Process between your InfoCube **B350DS##** and your DataSource **Z_SFLIGHT_##**. Use Extraction Mode “FULL” for the data transfer between your DataSource and your InfoCube.

Continued on next page

Task 7:

Create an InfoPackage for your DataSource in order to upload data into the PSA-table.

1. Create an InfoPackage with name **Flight data for GR##** for your DataSource **Z_SFLIGHT_##** in order to upload data into the PSA table.

Task 8:

Create a Process Chain to extract data for your InfoCube,

1. Create a Process Chain to extract data for your InfoCube **B350DS##**.

<i>Process Chain</i>	PC_GR##
<i>Long description</i>	Load flight data for GR##

2. Create a Start process in your Process Chain **PC_GR##**for scheduling purposes.

<i>Process Variants</i>	Start_GR##
<i>Long description</i>	Start Process GR##

3. Insert the Execute InfoPackage process into your Process Chain **PC_GR##** using your InfoPackage with name **Flight data for GR##**.
4. Load data into your InfoCube using the Process Chain you have just activated. After having started your Process Chain, use the Log view to follow-up the execution of the Process Chain. After successful execution of your Process Chain check the data in your InfoCube **B350DS##** using transaction LISTCUBE.

Solution 1: Overview of Data Transfer Process

Task 1:

Create an InfoCube and give it the name and description **B350DS##**.

1. Create the InfoCube **B350DS##** as a copy.

Field Name	Values
<i>InfoCube</i>	B350DS##
<i>Long Description</i>	GR## DTP Overview
<i>Copy From</i>	B350DS00

- a) From the menu, choose *Data Warehousing Workbench: Modeling* → *InfoProvider*.
- b) Expand the InfoArea tree by choosing *InfoProvider* → *BW_TRAINING* → *BW_CUSTOMER TRAINING* → *T_BW350*.
- c) Select your InfoArea *T_BW350_GR##*, right-click and choose *Create InfoCube*.

Field Name	Values
<i>InfoCube</i>	B350DS##
<i>Long Description</i>	GR## DTP Overview
<i>Copy From</i>	B350DS00

Choose the *Create*  icon.

Check the settings and fields in the InfoCube, but do not change anything.

Choose the *Activate*  icon.

Task 2:

Create a generic DataSource for the source system **T90CLNT090** using the table **SFLIGHT** as a basis.

1. Create a generic DataSource **Z_SFLIGHT##** of type “Transaction Data” for the source system **T90CLNT090**.

Continued on next page

<i>Applic. Component</i>	BW350 - ##
<i>Short description</i>	GR## Flight DS
<i>Medium description</i>	GR## Flight DataSource
<i>Long description</i>	GR## Flight DataSource
<i>View/Table</i>	SFLIGHT

- a) Use the menu *Data Warehousing Workbench: Modeling* → *Source Systems* and open the folder **BI** to find the source system **T90CLNT090**.
From the context menu of **T90CLNT090** choose the option *Customizing Extractors*.
- b) Use the menu *Data Transfer to the SAP Business Information Warehouse* → *Generic DataSources* → *Maintain Generic DataSources* and choose the *Execute*  icon.
In the field *Transaction Data* enter the value **Z_SFLIGHT_##** and choose the *Create*  icon.
- c) In the screen “Create DataSource for Transaction data: **Z_SFLIGHT_##**” enter the following values:

<i>Applic. Component</i>	BW350 - ##
<i>Short description</i>	GR## Flight DS
<i>Medium description</i>	GR## Flight DataSource
<i>Long description</i>	GR## Flight DataSource
<i>View/Table</i>	SFLIGHT

Choose the *Save*  icon .

In the dialog *Create Object Directory Entry* choose the *Local Object* button.

In the screen “DataSource: Customer version Edit” select the checkboxes for the fields *CARRID* and *CONNID* in column *Selection* to mark these fields as selection fields.

Choose the *Save*  icon .

Choose the *Back*  icon three times to return to BI and you should be in the Source Systems tree.

Continued on next page

Task 3:

You now need to replicate your new DataSource to BI.

1. Replicate your DataSource as a 3.x DataSource in BI system.
 - a) From the context menu of source system **T90CLNT090** choose *Display DataSource Tree* and find your application component **BW350-##**
 - b) From the context menu of your application component **BW350-##** choose *Replicate Metadata*.
 - c) In response to the dialog *Unknown DataSource* choose the radio button *as 3.x DataSource* and then click the Continue (Enter)  icon.

Task 4:

You discovered that you made a mistake by creating a 3.x DataSource, so you will have to migrate the 3.x DataSource to a BI DataSource.

1. Migrate your 3.x Datasource **Z_SFLIGHT_##** to a BI DataSource.
 - a) From the context menu of your DataSource **Z_SFLIGHT_##** choose *Migrate*.
 - b) You will be presented with the dialog *Migration of 3.x DataSource* and asked how you want to proceed.

In response to the dialog, choose *With Export*.



Note: When a DataSource 3.x (R3TR ISFS) is migrated, it is deleted and transferred to the DataSource (R3TR RSDS). The 3.x DataSource is deleted, along with the 3.x metadata object mapping (R3TR ISMP) and transfer structure (R3TR ISTS), which are dependent on it. If a PSA or InfoPackage (R3TR ISIP) already exist for the 3.x DataSource, they are transferred to the migrated DataSource, along with the requests that have already been loaded. The 3.x objects can also be exported as part of the migration process. This means it is possible to recover the metadata objects DataSource 3.x, the mapping and the transfer structure..

It is only possible to recover a DataSource 3.x if it was migrated With Export. This allows the objects DataSource 3.x (R3TR ISFS), mapping (R3TR ISMP), and transfer structure (R3TR ISTS) to recover the same status as they had before the migration.

- c) If you are presented with a logon screen for the BI system, enter your BI user id and password and choose *Enter*. This will allow BI to complete the replication of the DataSource's metadata as a BI DataSource.

Continued on next page

Task 5:

Create a Transformation between your Data Source **Z_SFLIGHT_##** and your InfoCube **B350DS##**.

1. From the context menu of your DataSource create a Transformation between your DataSource and your InfoCube

Object Type	InfoCube
Name	B350DS##

Use the following fields:

<i>CARRID</i>	CARR_ID
<i>CONNID</i>	CONN_ID
<i>FLDATE</i>	OCALDAY
<i>PRICE</i>	PRICE
<i>CURRENCY</i>	PRICE

Continued on next page

<i>PLANETYPE</i>	PLANETYPE
<i>SEATSMAX</i>	SEATSMAX
<i>SEATSOCC</i>	SEATOCC

- a) Use menu *Data Warehousing Workbench: Modeling* → *DataSources* → *BW Training* → *BW Customer Training* → *BW350 Extraction* → *Group ## Application Component* to find your DataSource **Z_SFLIGHT_##**.
- b) From the context menu of your DataSource choose *Create Transformation*.
- c) In the *Create Transformation* dialog box enter the following values:

<i>Object Type</i>	InfoCube
<i>Name</i>	B350DS##

Accept all other defaults and then click the *Transfer (Enter)*  icon.

- d) Map all fields as follows by dragging a line between the fields on the left and the fields on the right.

<i>CARRID</i>	CARR_ID
<i>CONNID</i>	CONN_ID
<i>FLDATE</i>	OCALDAY
<i>PRICE</i>	PRICE
<i>CURRENCY</i>	PRICE
<i>PLANETYPE</i>	PLANETYPE
<i>SEATSMAX</i>	SEATSMAX
<i>SEATSOCC</i>	SEATOCC

Activate the Transformation by choosing the *Activate*  icon.

Continued on next page

Task 6:

Create a Data Transfer Process between your InfoCube **B350DS##** and your DataSource **Z_SFLIGHT_##**.

1. Create a Data Transfer Process between your InfoCube **B350DS##** and your DataSource **Z_SFLIGHT_##**. Use Extraction Mode “FULL” for the data transfer between your DataSource and your InfoCube.

- a) From the menu, choose *Data Warehousing Workbench: Modeling* → *InfoProvider*.

Expand the InfoArea tree by choosing *InfoProvider* → *BW_TRAINING* → *BW_CUSTOMER TRAINING* → *BW350 Group##* and locate your InfoCube **B350DS##**.

- b) From the context menu of your InfoCube **B350DS##** choose *Create Data Transfer Process*.

In the dialog enter the following values:

<i>Object Type</i>	DataSource
<i>DataSource</i>	Z_SFLIGHT_##
<i>Source System</i>	T90CLNT090

Choose the *Continue (Enter)*  icon.

- c) On the *Extraction* tab, enter the following value from the table and leave the defaults:

<i>Extraction Mode</i>	Full
------------------------	-------------

- d) On the *Update* tab accept the defaults.
 - e) Activate the Data Transfer Process by choosing the *Activate*  icon.

Continued on next page

Task 7:

Create an InfoPackage for your DataSource in order to upload data into the PSA-table.

1. Create an InfoPackage with name **Flight data for GR##** for your DataSource **Z_SFLIGHT_##** in order to upload data into the PSA table.

- a) Expand the Data Transfer Process under your InfoCube until you see the DataSource for which you created the Data Transfer Process.

From the context menu of the DataSource choose *Create InfoPackage* and in the presented dialog enter the name **Flight data for GR##**.

- b) On the *Data Selection* tab you should see fields **CARRID** and **CONNID** available for selection criteria.
 - c) Go to the *Extraction* tab to make sure that the entry in the *Adapter* field is **Access to SAP Data through Service API**.
 - d) On the *Processing* tab you will see that the option **Only PSA** is not available for change.

→ **Note:** In BI, InfoPackages only extract data to the PSA table.

- e) On the *Data Target* tab, your InfoCube should be the only InfoProvider in the list.
 - f) On the *Update* tab the only option should be a *Full* update.
 - g) On the *Schedule* tab, do **not** schedule. Only save the InfoPackage by choosing the **Save** icon.

Task 8:

Create a Process Chain to extract data for your InfoCube,

1. Create a Process Chain to extract data for your InfoCube **B350DS##**.

Continued on next page

<i>Process Chain</i>	PC_GR##
<i>Long description</i>	Load flight data for GR##

- a) Either use transaction RSPC or choose the *Process Chains*  icon.
- b) In the *Process Chain Display Planning View* choose the *Create*  icon.
In the *New Process Chain* dialog, enter the values:

<i>Process Chain</i>	PC_GR##
<i>Long description</i>	Load flight data for GR##

then choose the *Continue (Enter)*  icon.

2. Create a Start process in your Process Chain **PC_GR##** for scheduling purposes.

<i>Process Variants</i>	Start_GR##
<i>Long description</i>	Start Process GR##

- a) When you are presented with the dialog to create a Start process.
Click the *Create*  icon.
Enter the following values:

<i>Process Variants</i>	Start_GR##
<i>Long description</i>	Start Process GR##

- b) Choose the *Continue (Enter)*  icon.

In the *Maintain Start Process* screen, make sure that the Scheduling Option is set to **Direct Scheduling** which should be the default and choose the *Change Selections* button.

In the *Start Time* dialog choose the *Immediate* button and then the *Save*  icon.

In the *Maintain Start Process* screen, choose the *Save*  icon.

Choose the *Back*  icon and then the *Continue (Enter)*  icon.

Continued on next page

3. Insert the Execute InfoPackage process into your Process Chain **PC_GR##** using your InfoPackage with name **Flight data for GR##**.
 - a) To access other processes choose the *Process Type* icon  located just above the Navigation block *Descript./Descript..* This will open a list of folders which contains the allowed processes for a Process Chain.
 - b) Open the folder *Load Process and Post Processing* and find the process *Execute InfoPackage*. Using the mouse, select and then drag and drop this process over to the work area for your Process Chain.
 - c) In the dialog *Insert Execute InfoPackage* find and choose your InfoPackage **Flight data for GR##** so that it is inserted into the field *Load Data*. The BI system will insert the system-generated technical name into the field.

Choose the *Continue (Enter)*  icon.

The system will insert not only the Execute InfoPackage process but also the Data Transfer Process.

- d) Connect the Start process with the Execute InfoPackage process by dragging a link between them by clicking on the Start process and holding the primary mouse button down and then drag to the Execute InfoPackage process and release the mouse button.
- e) Check your Process Chain by switching to the Checking View using the menu *Goto → Checking View*. All the processes in your Process chain should be green in color indicating that your Process Chain is executable.

Finally activate your Process Chain using the *Activate*  icon.

Continued on next page

4. Load data into your InfoCube using the Process Chain you have just activated. After having started your Process Chain, use the Log view to follow-up the execution of the Process Chain. After successful execution of your Process Chain check the data in your InfoCube **B350DS##** using transaction LISTCUBE.
 - a) Execute your Process Chain by choosing the *Execute*  icon.
 - b) Switch to the Log view using the menu *Goto → Log View*
In the *Date Selection* dialog, choose the radio button *Today* and then choose the *Transfer (Enter)*  icon.
You may need to choose the *Refresh*  icon several times until the Process Chain is finished.
 - c) Check the data in your InfoCube by using the transaction LISTCUBE and enter the technical name of your InfoCube and choose the *Execute*  icon.
On the following screen choose the button *Fld Selectn for Output* and choose several Characteristics. The Key Figures are selected by default. then choose the *Execute*  icon and you will be returned to the previous screen.
Choose the *Execute*  icon again to see the data in your InfoCube.



Lesson Summary

You should now be able to:

- Name the different source systems and source system types
- Explain the importance of DataSources for data acquisition
- Describe the various transfer mechanisms

Lesson: BI Content

Lesson Overview

This lesson introduces BI Content, focusing on its areas of application and components or objects. The procedure for transferring BI Content objects is also explained.



Lesson Objectives

After completing this lesson, you will be able to:

- Name BI Content (Business Content) objects and areas
- Explain the object versions
- Describe the transfer process for BI Content

Business Example

After analyzing reporting requirements and creating the data model, your company wants to set up SAP BW. Your task is to check BI Content to see which BI Content objects can be used for data warehousing and which additional settings need to be made to meet your particular requirements.

BI Content

BI Content is supplied as an add-on for SAP BW. The naming convention is **BI Content 3.5.3 Add-On** (or similar, depending on the respective release).

→ **Note:** The conventional term **Business Content** will be used from here on.

With Business Content, role and task-related information models are prepared on the basis of consistent metadata. These are used amongst other things to support the entire Data Warehouse process, from data extraction through to data analysis. These information models include numerous preconfigured objects that simplify and speed up the implementation of SAP BW. The following graphic shows a selection of these objects.

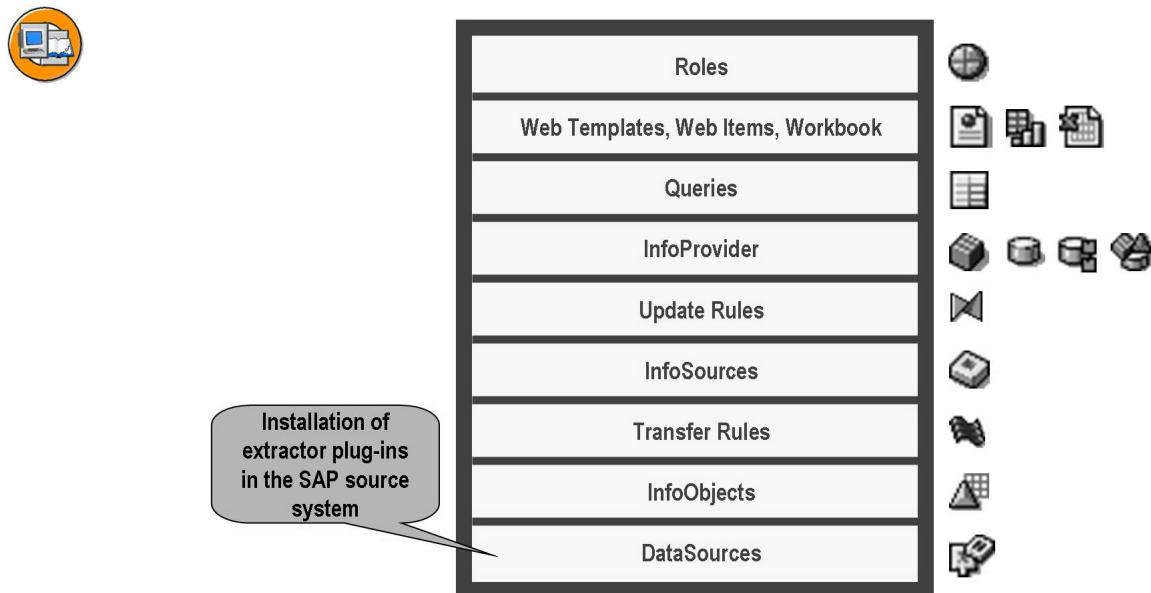


Figure 12: Business Content: Objects



Hint: If necessary, Business Content for SAP source systems is installed using extractor plug-ins (add-on components). These create the technical preconditions for the extraction. This aspect is dealt with in detail in unit 2 *Data Extraction from SAP Source System*.

In close cooperation with the customer, SAP continually enhances *Business Content* with additional information models. Depending on the customer's requirements, these can

- be used without adaptations
- be adapted by enhancements
- serve as a template or example for customer-specific objects

Business Content: Areas

Business Content objects are grouped in different areas to enable you to use them in the best possible way. The following graphic shows a selection of these areas.



- **Analytical Applications**
- **Customer Relationship Management (CRM)**
- **Supplier Relationship Management (SRM)**
- **Supply Chain Management (SCM)**
- **Product Lifecycle Management (PLM)**
- **Financials**
- **Human Resources**
- **Industry Solutions** (Automotive, Chemicals, Retail, Utilities, etc.)
- **Computing Center Management System (CCMS)**
- **Non-SAP Sources** (Oracle Content, D&B Purchasing, etc)
- **Demo Content**

Figure 13: Business Content: Areas



Note: Demo content available for test and training purposes. This contains predefined objects (such as queries and BasicCubes) with sample data. The sample data is supplied with flat files that are also included in the demo content. The technical name of all the demo content objects begins with OD_.

Object Versions

The following graphic shows the object versions and how the versions are used in the delivery process.



Version	Meaning
D(elivery)	SAP standard version (BI Content version)
A(ctive)	Active version
M(odified)	Modified version

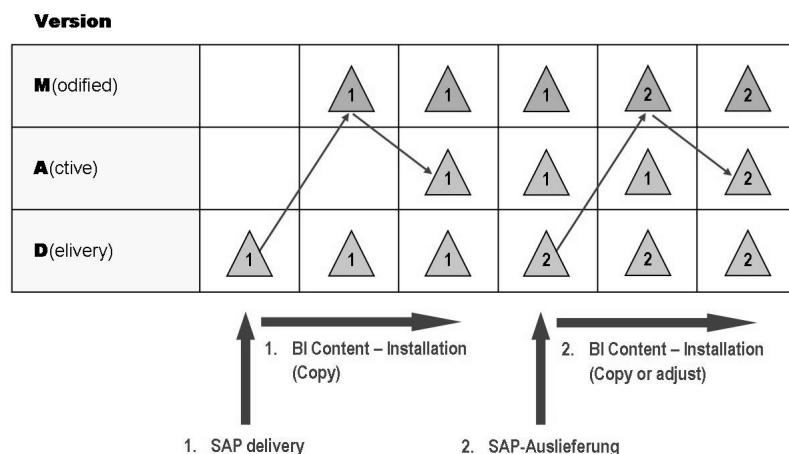


Figure 14: Object Versions

The object versions illustrated in the above graphic are explained below:

After you have installed the Business Content as an add-on component in the SAP BW system, all the SAP Business Content objects are available to you in the **D(elivered) version**. To work with these objects, you have to transfer them from the Business Content. When you transfer Business Content objects, the contents of the D version are copied and transferred to an **A(ctive) version** and corresponding objects are generated in the ABAP Dictionary and programs. The D version is not deleted after the transfer. This means that the contents can be transferred more than once, and transferred Business Content objects can be changed or deleted without this affecting the D version. If an activated Business Content object is changed to adapt it to a customer's specific requirements, a new, **M(odified) version** is generated. You can then activate this version, whereupon it overwrites the older, active version.

Only objects that have been saved as A-versions are exported from the development system. When they are imported into the target system, these objects are imported either directly into the A version or into an M version, depending on the object type.

When you upgrade Business Content, SAP supplies the Business Content objects in a new D version. When you transfer the contents again, you can compare the Business Content objects with the existing objects in the A version. This is necessary when the A version of a Business Content object is not identical with the new D version (because it was changed, or because a more recent version of

the Business Content exists, for example). When comparing the objects, you can decide which properties of the new Business Content object should be transferred and which properties of the active object should be retained.

**Hint:**

1. You can change objects that were transferred from the Business Content, since only the derived A version and not the Business Content version of the objects will be changed.
2. Not all objects can be compared!
3. If you do not compare the objects, the new version is transferred (copied) and the (old) A version overwritten when you activate the Business Content.

Transferring Business Content Objects

The transfer process for Business Content objects can be described in three stages:



(1) Choose Selection Level

(2) Select Business Content objects with grouping method

(3) Transfer Business Content objects

Figure 15: Business Content: Transfer Process

The transfer process is performed in the *Business Content* function area of the *Administrator Workbench*.



Figure 16: Administrator Workbench: Business Content

The following section describes in greater detail the three steps referred to in the above graphic:

(1) Choose the selection level

In the navigation window to the left (in the above graphic), choose the selection level:

- *InfoProviders by InfoAreas*
- *InfoObjects by Application Components*
- *InfoSources by Application Components*
- *Roles*
- *Object Types*
- *Objects in BW Patch*
- *Transport Request*
- *Packages*

When you have chosen the selection level, the corresponding Business Content objects are displayed in the navigation window in the middle (in the above graphic).

(2) Select Business Content objects with grouping method

Before you select the objects to be activated in the middle navigation window, you can choose the grouping method from the *grouping* selection list. With the grouping method you determine the (dependent) objects to be additionally included when activating the Business Content objects. The following options are available:

- *Only necessary objects* (default setting)
Only those objects that are essential for activating the selected objects are included (minimal selection).
- *Data flow before*
All objects are collected that supply data to the selected objects.
- *Data flow after*
The objects that receive data from the selected objects are collected.
- *Data flow before and after*
All the objects are collected that both supply data to and receive data from the selected objects.
- *Backup for system copy*
With this setting you can collect objects for a transport request. This request can be re-imported after a system copy.

After you have selected the grouping method you can transfer the objects to be activated in the middle navigation window by *drag & drop* to the *Collected Objects* navigation window on the right.

**Hint:**

1. In the *Collection Mode* selection list you can determine how you want to collect the objects:
 - *Collect automatically* (default setting)
The (dependent) objects are collected directly when the objects to be activated are selected.
 - *Collect manually*
The (dependent) objects are then only collected when you choose *collect dependent objects*.
2. When transferring Business Content objects you have the option of assigning source systems. You make this assignment using *assign source systems* (in the graphic above). By making this assignment you determine the source systems in which these objects are collected for transfer. You should only ever choose the required source systems, as this could otherwise lead to unnecessarily long waiting periods. The assignment of source systems is only relevant for source system-dependent objects (such as transfer rules and InfoPackages). If several source systems are available, only those objects assigned to the selected source systems are collected for transfer. Objects that are not assigned to the selected source systems are ignored. If you do not select a source system, all source systems are automatically assigned.

(3) Transfer Business Content objects

Before you activate the collected objects in the *Collected Objects* navigation window on the right, check the following columns in this window. (To keep the illustration simple, these are not shown in the above graphic):

- *Save (Enter)*

The following Business Content objects are flagged by default in this column:

- Objects that were transferred for the first time
- Objects that were supplied again by the Business Content in a more recent version

- *Compare or Copy*

A checkbox is displayed in this column if a comparison between the Business Content version (D version) and the A version is possible. The indicator is set by default. In this case, your version is compared with the new D version. If you delete the indicator the new D version is copied in full. This means that the existing A version is overwritten and thus no longer available.

After you have made the checks described above, you can select the following options from the *Transfer* selection list:

- *Simulate transfer*
- *Save (Enter)*

The objects are transferred immediately.

- *Transfer in the background*

The object transfer is scheduled.

- *Transfer and transport*

The objects are transferred immediately and written to a transport request.



Hint:

1. You can use Business Content DataSources from an SAP source system in SAP BW only if you have transferred them from the Business Content in the SAP source system and replicated them in the SAP BW system (see the unit Data Extraction from SAP Source Systems).
2. You can display all the Business Content objects in the metadata repository for the Administrator Workbench. Furthermore, you can display the data flow for a Business Content in the metadata repository. This is useful if you want to assess its complexity with regard to the transfer. In the metadata repository, choose: Business Content → Object: for example, the ODS object orders (0SD_03) → Network display of the data flow

3. You can also develop a **customer or partner content**. As of SAP BW 3.5 documentation is available for developing a customer or partner content.

The customer or partner content functions enhance and extend the options for using the Business Content supplied by SAP.

4. To supply SAP Business Content as well as develop your own content and supply this as customer or partner content, you need at least one SAP BW system to develop the content. If you want to develop DataSources and extractors, again you will need an SAP source system to develop the content. We recommend you connect a content test system to the content development system.

A content development system is an SAP system or SAP BW system with the system setting set to content. To use a development system as a content system you need to set the relevant indicator for the maintenance view *RSADMINSV* using transaction *SM 31*.

5. The **technical content** is available for SAP BW statistics. You do not need to import this content as an add-on. It is an integral part of the SAP BW system. The technical content is transferred in two steps:
 - Transfer the technical content for the extraction
 - Transfer the technical content for SAP BW objects



Lesson Summary

You should now be able to:

- Name BI Content (Business Content) objects and areas
- Explain the object versions
- Describe the transfer process for BI Content



Unit Summary

You should now be able to:

- Describe the concept behind SAP NetWeaver 2004s
- List the advantages and benefits of SAP NetWeaver 2004s
- Acquire an overview of the SAP NetWeaver 2004s architecture and its components
- Describe the position of BI in SAP NetWeaver 2004s
- Describe the BI architecture and its functional areas
- Name the different source systems and source system types
- Explain the importance of DataSources for data acquisition
- Describe the various transfer mechanisms
- Name BI Content (Business Content) objects and areas
- Explain the object versions
- Describe the transfer process for BI Content

Unit 2

Data Flow in Data Acquisition

Unit Overview

This unit begins with an overview of the flow of data during data acquisition. It discusses the functions used to transform data during extraction and how to access source data directly without data extraction as well as real time data acquisition.



Unit Objectives

After completing this unit, you will be able to:

- Describe the data transfer process
- Contrast the new data flow with the previous data flow
- Describe the conversion path to the new data transfer process
- Describe the transformation process
- List the functions available for transformation of data during extraction
- Describe how to access data in BI source systems directly
- Describe the real time data acquisition process
- Describe real time data acquisition with the Service API
- Describe real time data acquisition with web services

Unit Contents

Lesson: Data Flow Overview in BI.....	54
Lesson: Transformation Process	79
Procedure: Creating Variables for Quantity Conversion	96
Procedure: Creating Quantity Conversion Types	97
Exercise 2: Data Acquisition Transformation in BI	99
Lesson: Direct Access to Source System Data.....	110
Exercise 3: Direct Access for Master Data.....	117
Lesson: Real Time Data Acquisition	125
Procedure: Define the data model for real-time data acquisition.....	128
Procedure: Creating an InfoPackage for Real-Time Data Acquisition	136
Procedure: Creating a Data Transfer Process for Real-Time Data Acquisition	138

Lesson: Data Flow Overview in BI

Lesson Overview

This lesson describes the flow of data during transfer and explains how to convert to the latest data transfer process.



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the data transfer process
- Contrast the new data flow with the previous data flow
- Describe the conversion path to the new data transfer process

Business Example

Your company would like to set up reporting using sales and purchasing data from your BI source system. You have previously extracted this data and would like to convert to the new data transfer process.

Data Flow During Data Transfer

Source Systems

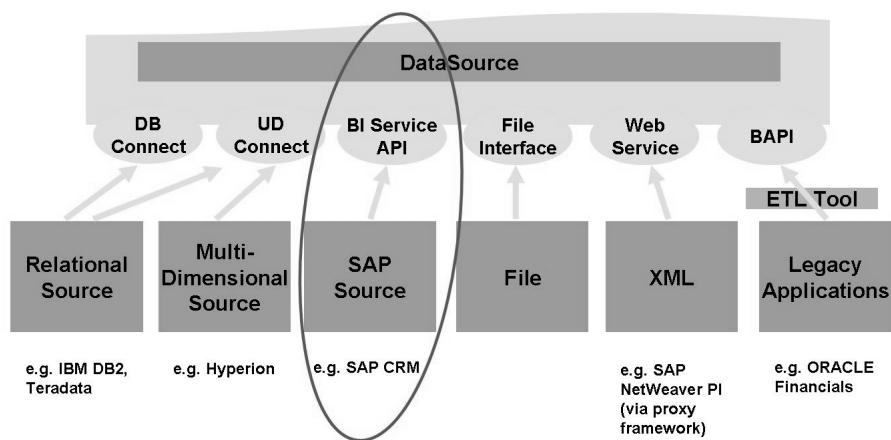


Figure 17: Data Sources

DataSource

As of SAP NetWeaver 2004s, an updated object concept is available for DataSources. It is used in conjunction with the changed objects concepts in data flow and process design (transformation, InfoPackage for loading to the PSA, data transfer process for data distribution within BI). In BI in the DataSource

maintenance, you determine which DataSource fields contain the information that is relevant for decision making for a business process and should be transferred. Upon activation of the DataSource, a PSA table is generated in the input layer of BI and data can then be loaded.

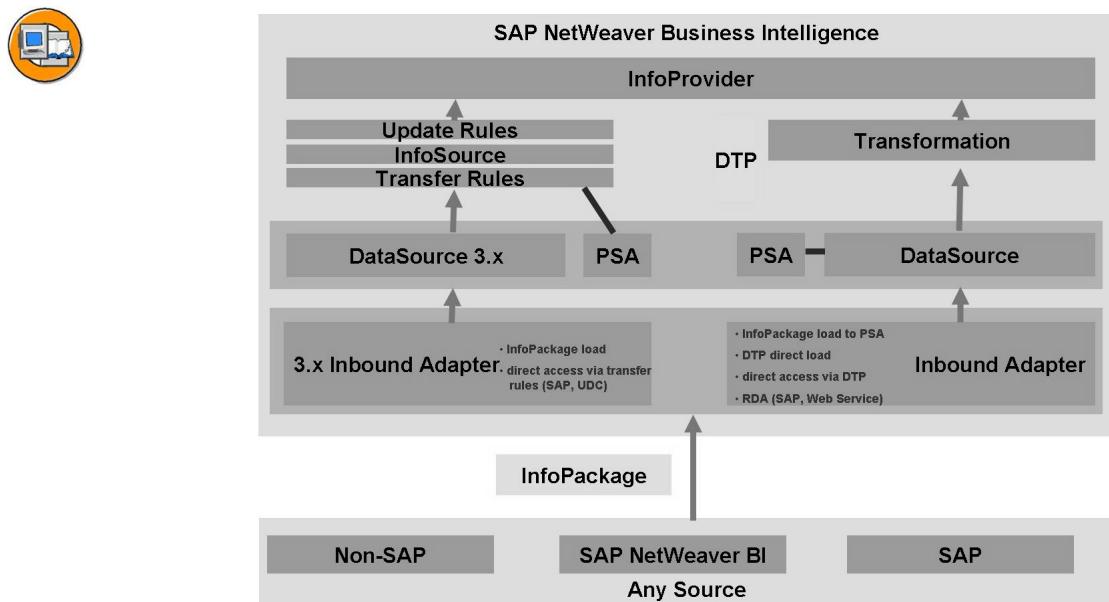


Figure 18: DataSource 3.x XOR “New” DataSource

In the transformation, you determine what the assignment of fields from the DataSource to InfoObjects from BI should look like. Data transfer processes facilitate the further distribution of the data from the PSA to other targets. The rules that you set in the transformation apply here.

The graphic below 'DataSource: Data Acquisition in BI' highlights the role played by the DataSource in data acquisition in BI as the central object for data acquisition in BI:

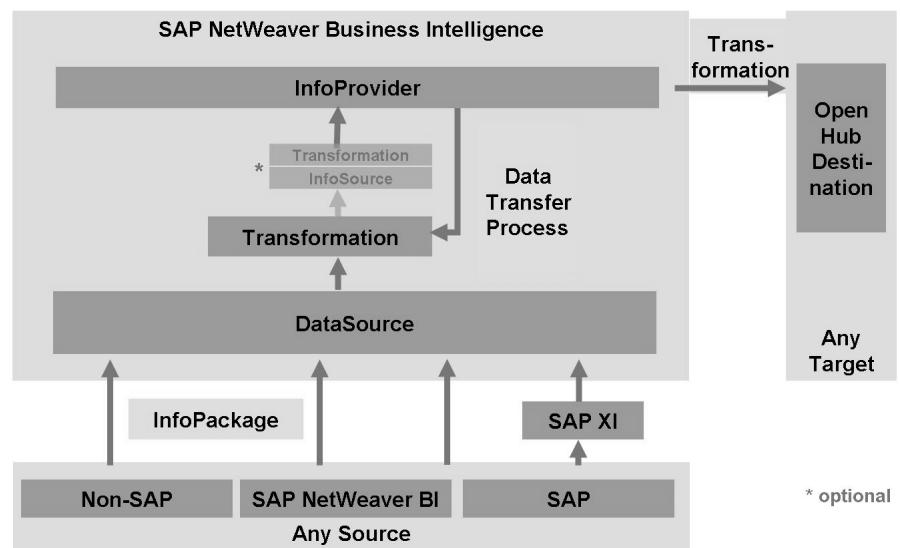


Figure 19: DataSource: Data Acquisition in BI

Transformation

The transformation process allows you to consolidate, cleanse, and integrate data. You can semantically synchronize data from heterogeneous sources. When you load data from one BI object into a further BI object, the data is passed through a transformation. A transformation converts the fields of the source into the format of the target.

You create a transformation between a source and a target. The BI objects DataSource, InfoSource, DataStore object, InfoCube, InfoObject and InfoSet serve as source objects. The BI objects InfoSource, InfoObject, DataStore object and InfoCube serve as target objects. The following graphic illustrates how the transformation is integrated in the dataflow:

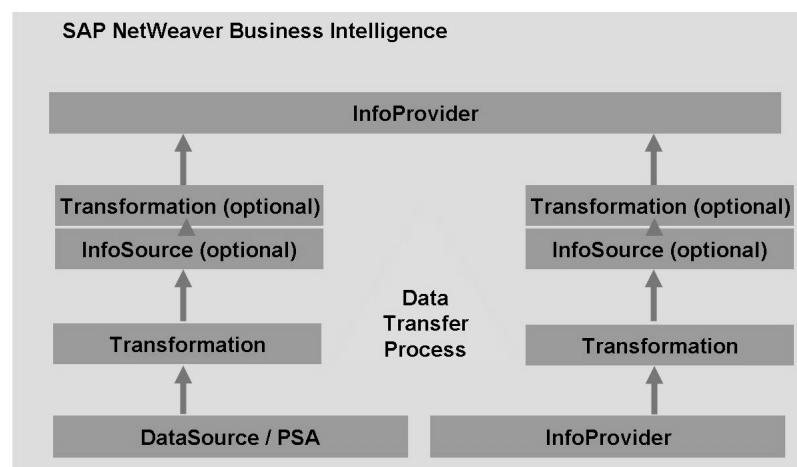


Figure 20: Data Transfer Process

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 to highly complex transformations. More detail on transformations are included in the lesson, “Transformation Process”

InfoSource

An InfoSource is a non-persistent structure consisting of InfoObjects for joining two transformations. You use InfoSources if you want to run two sequential transformations in the data flow without storing the data again. If you do not have transformations that run sequentially, you can model the data flow without InfoSources.



In this scenario, the InfoSource is used as

- Uniform source for several targets
- Target from different sources

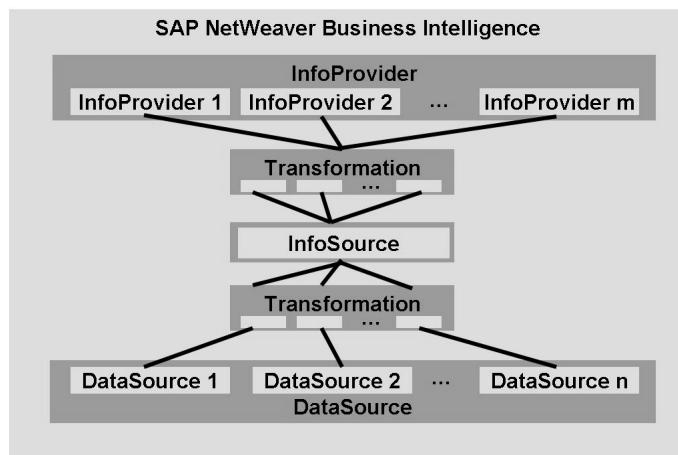


Figure 21: InfoSource

InfoSources should be used in the following scenarios:

- If you want to execute two transformations before data is written to the InfoProvider, you need an InfoSource to join these two transformations.
- If you want to update data from several sources (more than one DataSource for one or more source systems), it is useful to consolidate the data in an initial transformation at InfoObject level and in a second transformation, perform transformations that are independent of the data and are valid for all data. This allows you to react in a flexible way to changes that are not Data Source dependent.

Data Acquisition Using DataSource 3.x, InfoSource 3.x and Update rules 3.x

Transfer of data in SAP BI 3.x utilized an object called DataSource 3.x below. Data transfer was only possible when the DataSource 3.x was assigned to an InfoSource 3.x and the fields of the DataSource 3.x in the transfer structure maintenance InfoObjects were assigned to an InfoSource 3.x.

DataSource 3.x

Until now DataSources have been known in the BI system as replicated DataSources. When the transfer rules 3.x are activated, and thus the transfer structure 3.x, a PSA table is generated into which the data can be loaded.



Hint: A new type of InfoSource is available as of Release SAP NetWeaver 2004s. You can continue to create and use 3.x InfoSources, however, we recommend that you use the new InfoSource concept with the new transformation concept.

In the Data Warehousing Workbench, an icon before the description identifies an object that is available for the new concept.

The following graphic provides an example of this with a flat file DataSource and a DataSource that has been replicated from the SAP source system.

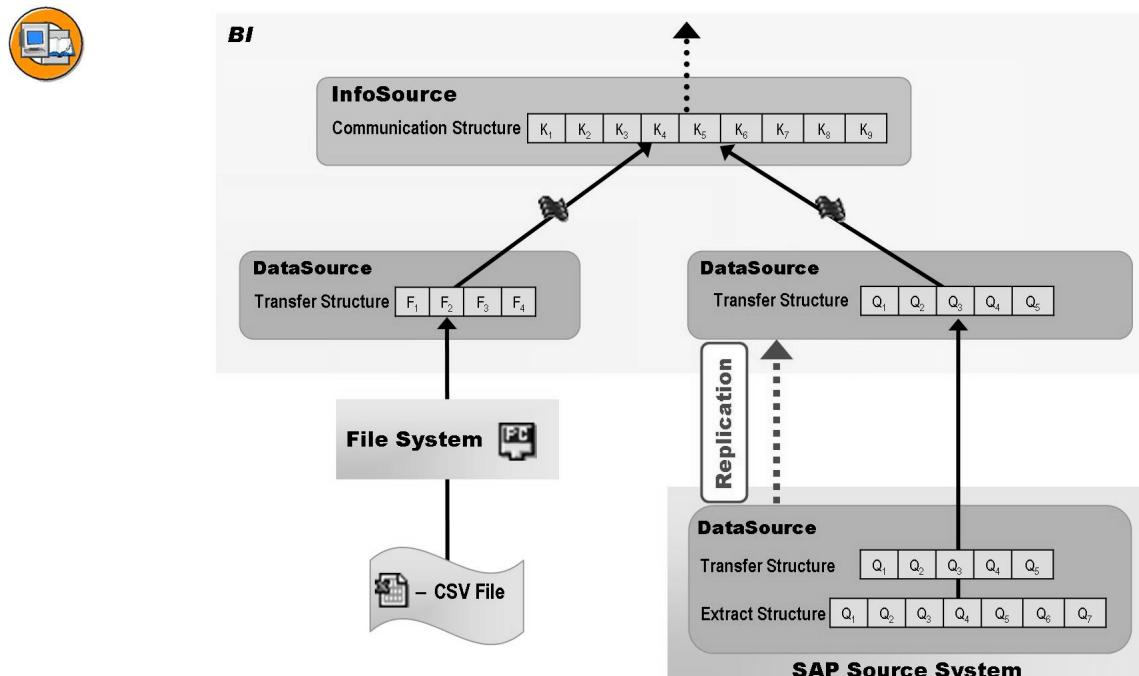


Figure 22: DataSource 3.x: Metadata in BI

For DataSource 3.x, the transfer structure is used to move the extracted data to the communication structure with the transformation. Activating the transfer rules in BI generates the transfer structure. With an SAP source system, the transfer structure is also generated in the SAP source system (metadata download).

If your dataflow is modeled with objects based on the BI 3.x concept (InfoSource 3.x, transfer rules 3.x, update rules 3.x) and the process design is based on these objects, you can continue to work with DataSources 3.x to transfer data into BI from a source system.

InfoSource 3.x

An InfoSource 3.x is a quantity of logically related InfoObjects, and stages consolidated and transformed data for updating to the BI data target.

A DataSource 3.x is assigned to an InfoSource 3.x in BI. If logically related fields exist from different source systems, these fields can be combined to form an InfoSource 3.x in BI by assigning several DataSources 3.x to an InfoSource 3.x.

The type of data (transaction data or master data) that can be updated and the data targets that can be updated depend on the type of InfoSource 3.x. A distinction is made between the two InfoSource 3.x types:

- *InfoSource with Direct Update*
(Master data attributes/ texts/hierarchies)
- *InfoSource with Flexible Update*
(Master data attributes/ texts/transaction data)

From a technical point of view, an InfoSource 3.x comprises one or more communication structures, depending on the InfoSource 3.x type.

InfoSource 3.x with Direct Update

With an *InfoSource with Direct Update*, master data (attributes, text data or hierarchies) are updated directly (in other words, without update rules 3.x) to the corresponding master data tables for an characteristic InfoSource 3.x after leaving the corresponding communication structure. You have to assign the characteristic InfoObject to an application component in the maintenance screen for the characteristic InfoObject (Master Data/Texts tab page) or in the InfoSource tree using the context menu for an application component.

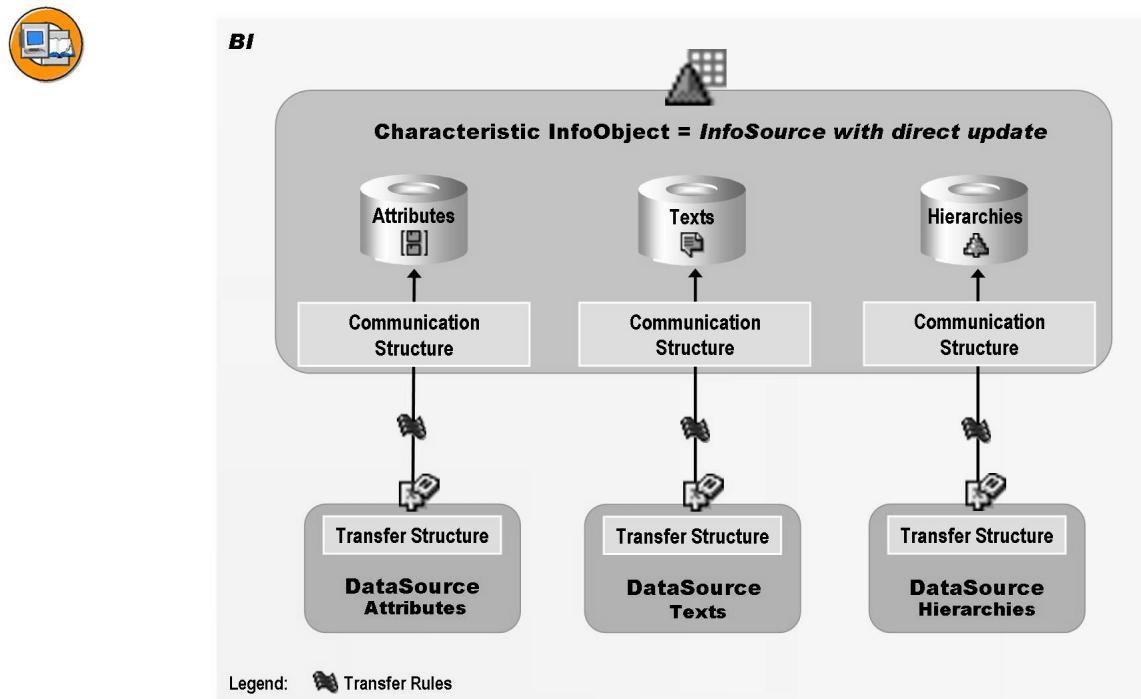


Figure 23: InfoSource with Direct Update



Hint: For direct InfoSources (master data updates), there is no difference between the SAP BI 3.x and the SAP NetWeaver 2004s InfoSource. The only prerequisite is that the InfoObject is defined as an InfoProvider.

From a technical point of view, this InfoSource 3.x type includes logically related field that are offered in three flat structures as an option, the so-called communication structures (for attributes/texts/hierarchies). These structures serve to update the data to the master data tables for the characteristic InfoObject.

**Note:**

1. Whether a communication structure exists for hierarchies that enables you to define transfer rules does not depend directly on the InfoSource, but on the 'hierarchy' type DataSource assigned to it. If, in addition to the IDoc transfer method, this DataSource supports the PSA transfer method, and you select the latter from the maintenance screen for the transfer rules, the communication structure is generated, allowing you to define transfer rules. This is possible for 'hierarchy' type flat file DataSource, for example.
2. In the following cases, it is not possible to use a characteristic InfoObject as an InfoSource with direct update:
 - In combination with the characteristic InfoObject OSOURSYSTEM (source system ID).
 - The InfoObject is a unit InfoObject (for example, 0UNIT, 0CURRENCY)
 - It has neither attributes, texts nor hierarchies

InfoSource3.x with Flexible Update

With an *InfoSource with flexible Update*, update rules are used to update the data from the communication structure to the data target assigned to it (Standard InfoCube, DataStore object or characteristic InfoObject). Several data targets can be supplied from this kind of InfoSource. You can use this InfoSource to update master data (attributes/texts) and transaction data. Similar to an *InfoSource with Direct Update*, this InfoSource type is created in the InfoSource tree in the Data Warehousing Workbench, using the context menu for an application component.



Caution: Hierarchies cannot be updated using an *InfoSource with Flexible Update!*

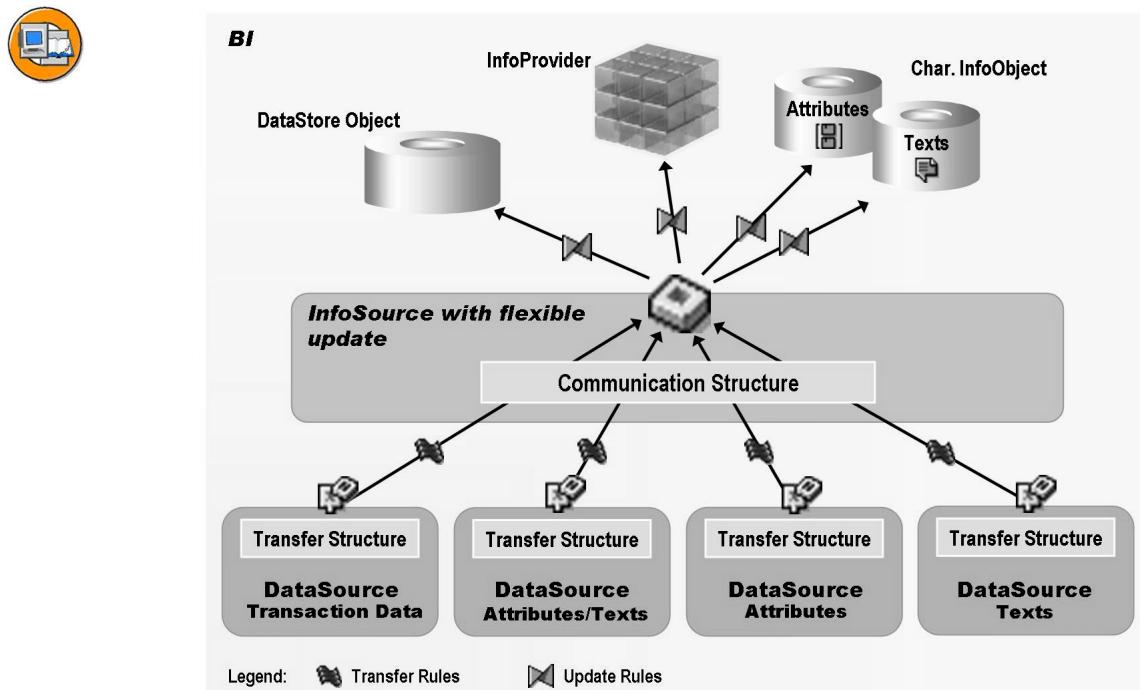


Figure 24: InfoSource with Flexible Update

The following update options are available for this InfoSource type:

- Master data (attributes/texts) and transaction data can be stored in DataStore objects.
- You can use update rules to update data from DataStore objects to other DataStore objects, Standard InfoCubes or master data tables (for attributes/texts) for characteristic InfoObjects.
- Of course, it is also possible to update transaction data to Standard InfoCubes, or from master data (attributes/texts) to the corresponding master data tables for the characteristic InfoObject without using DataStore objects.

From a technical perspective, this InfoSource type includes logically related fields that are offered in a single communication structure for updating to data targets.



Note:

1. When you use an *InfoSource with Flexible Update* it is not always immediately obvious whether the data is master data or transaction data. It is therefore a good idea to indicate this in the name of the InfoSource.
2. With an *InfoSource with Flexible Update* you can check the data loading process for referential integrity in the maintenance screen for the relevant communication structure.

In BI you have several options for transforming data for the acquisition process:

- *Transfer rules*
- *Update rules*

Transfer rules

Transfer rules are defined in BI for harmonizing or cleaning up data from the transfer structure to the communication structure. The prerequisite is that one or more DataSources are assigned to an InfoSource, though the transfer rules are defined for each DataSource. This means that transfer rules are DataSource-specific, and consequently source system-specific.

You get to the maintenance screen for the transfer rules using the assignment that you have made in the context menu for the Data Warehousing Workbench. With the transfer rules, you determine which InfoObjects in the InfoSource (fields for the communication structure) are filled from which field in the transfer structure, and the method that is used in the process.

- *Mapping*

With mapping you determine which field in the DataSource or transfer structure is assigned to which InfoObject in the InfoSource.

- *Transfer types*

The transfer types determine how data is transferred from the transfer structure to the communication structure:

- Data is transferred on a *1:1* basis, in other words, it is not modified
- Fields in the communication structure can be supplied by a *constant*
- Fields in the communication structure can retain their *initial* value
- Creating *formulas*
- Creating *ABAP routines*



Hint:

1. Local transfer routine:

When maintaining the transfer rules from the transfer structure to the communication structure, you have the option of assigning a transfer routine to an InfoObject. This transfer routine applies locally (source system-specific) for the InfoObject.

2. Global transfer routine:

You can create a transfer routine in the maintenance screen for the characteristic InfoObject. This applies globally (across all source systems) for the characteristic InfoObject, and is included in all the transfer rules. If data is loaded from different source systems to

the BI system, for example, you only need to maintain the global transfer routine once (in contrast to the local transfer routine). The use of global transfer routines is useful if a characteristic InfoObject is always transformed with the same ABAP routine, as is the case with key harmonization or conversion checks, for example.

3. If you have created a local and a global transfer routine, both are used: First the local and then the global transfer routine.
4. if you load the data using the PSA table (see under *Transfer Method*), you have the option of creating a **start routine** in the maintenance screen for the transfer rules. This is run for each data package after the data is written to the PSA table and before the transfer rules are executed. With the start routine you can change (for example, add or delete data) or check data packages. It is not possible to use start routines for transfer rules for the 'IDoc' transfer method.

The following graphic provides another summary of the most important statements regarding transfer rules:

The following graphic shows the maintenance screen for transfer rules:

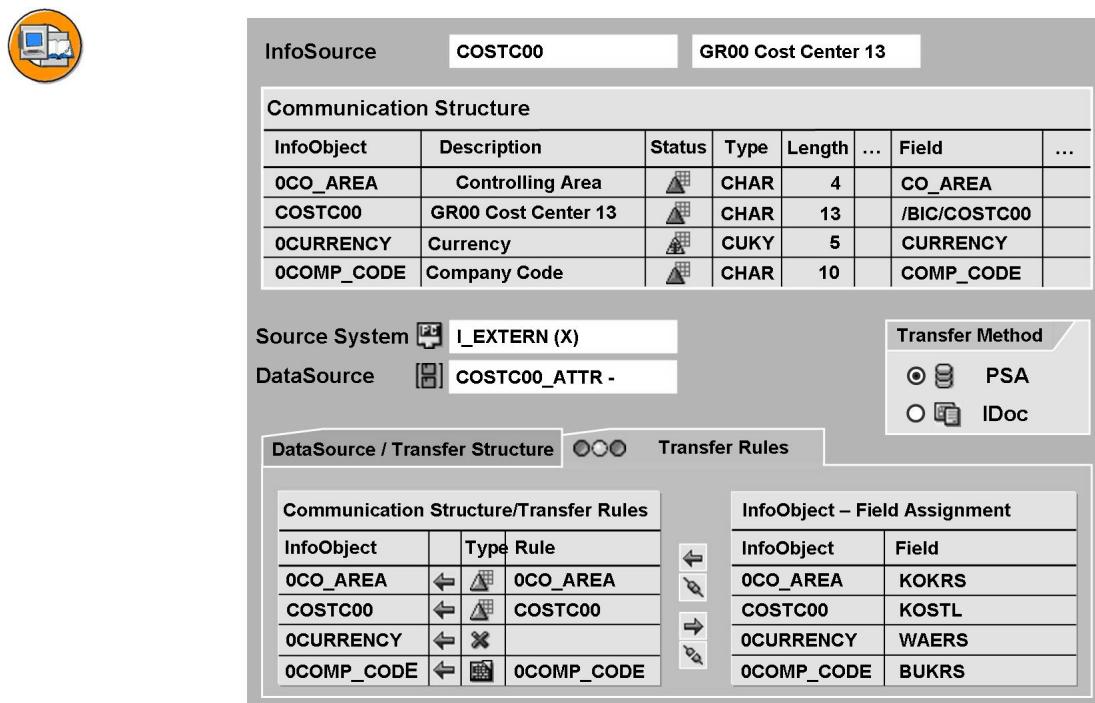


Figure 25: Defining Transfer Rules

When defining transfer rules you can choose between the following two **transfer methods**:

- *PSA (Persistent Staging Area)*
- *IDoc (Intermediate Document)*

The *transfer method* only determines how the data is transferred. Both methods are compared in the following table:



PSA (= Persistent Staging Area)	IDoc (= Intermediate Document)
Maximum data record length: 1962 byte Number of fields per data record restricted to 255	Maximum data record length: 1000 byte
Uses tRFC as transfer log	Uses tRFC as transfer log
Data loading process triggered by a <i>request IDoc</i> (= special <i>Info-IDoc</i>)	Data loading process triggered by a <i>request IDoc</i> (= special <i>Info-IDoc</i>)
Uses <i>Info-IDocs</i> , with which the monitor for overseeing the data load process is updated	<ul style="list-style-type: none"> - Uses Info-IDocs, with which the monitor for overseeing the data load process is updated - Use of data IDocs
Advantages: <ul style="list-style-type: none"> - Improved data load performance as larger data packages can be transferred - Optional: PSA tables can be used as inbound storage locations as this allows error handling in BI to be used Disadvantage: Restricted use of hierarchy upload	Advantages: <ul style="list-style-type: none"> - Detailed logs created by control record and status record in data IDoc - Hierarchy upload Disadvantage: No error handling in BI

Figure 26: Transfer Methods

The comparison highlights why the *PSA* transfer method is recommended. With the exception of 'hierarchy' type BI Content DataSources, all the others (including generic DataSources such as 'hierarchy' type flat file DataSources) support the *PSA* transfer method.

If you select the *PSA* transfer method, the system generates a **PSA table** for each DataSource in BI, in addition to the transfer structure, when you activate the transfer rules. The structure of a PSA table corresponds with that of the

transfer structure, except for the key fields. The key for a PSA table is made up of a *request*, *data package* and *data record number*. The extracted data can be temporarily stored in the PSA table before it is run through the transfer rules.

**Hint:**

1. With the *IDoc* transfer method you have to make sure that the data stores in the ALE inbound and outbound processing is emptied or reorganized.
2. With the *PSA* transfer method you have to make sure that you manage both the *PSA tables* (providing these are used), and the data stores in the ALE inbound and outbound processing for the *Info IDocs*.

For the *PSA* transfer method, other options (see the following graphic) are also available for updating data with regard to the PSA tables in the scheduler (Maintain InfoPackage) that are explained in the following section.

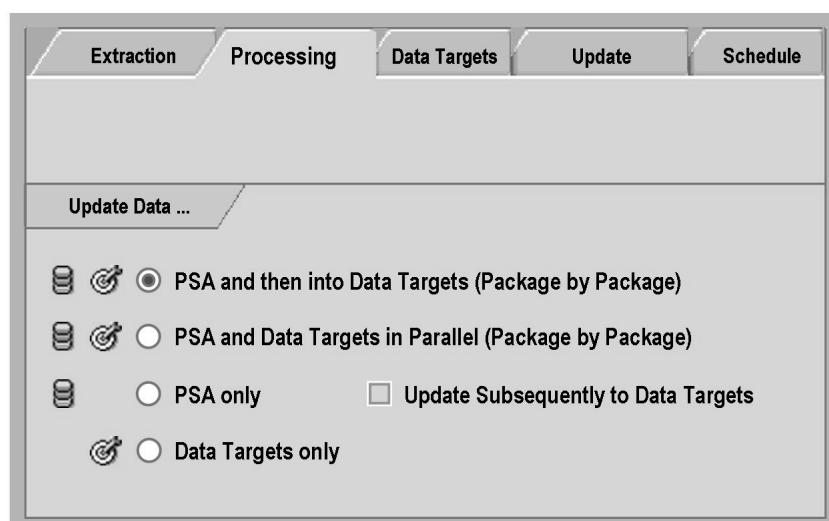


Figure 27: Scheduler (Maintain InfoPackage): Update Options

- **PSA and then data targets (by package)**

With this update type, a dialog process is started for each data package in a request that writes the data package to the PSA table. As soon as the data package has been successfully transferred to the PSA table, the data is updated to the data targets using the same process. The process is only

available for updating other data packages to the PSA table again when the data update from the previous package to both the PSA table and the data target has been completed.

**Hint:**

1. In the BI system, the maximum number of dialog processes required for each data request corresponds with the setting you have made in the transaction *SBIW* (Customizing the Extractors) for the maximum number of parallel processes in the source system.
2. If errors occur when posting data to the data targets, you can update the data packages from the PSA table to the data targets again after you have eliminated the errors. If the update to the data targets was terminated in the transfer or update rules, you can use the transfer and update rule debug option to search for the error. You can also simulate the update.
3. You can use error handling ('Update' tab page in the InfoPackage) to determine how the system should respond to data with errors.

- **PSA and data targets in parallel (by package)**

With this update type, a dialog process is started for each data package in a request that writes the data package to the PSA table. As soon as the data package has successfully been updated to the PSA table, a second, parallel process is started that writes the data package to the data targets (in contrast to the update type above). In this way, the first dialog process (update to the PSA table) is already available to update another data package to the PSA table, even though the second dialog process (update to data targets) is still updating the data package to the data targets.

**Hint:**

1. The maximum number of parallel processes set in the source system in transaction SBIW does not limit the number of processes in the BI system. You may therefore need many dialog processes in the BI system to update the data. You should therefore ensure that there are enough dialog processes available in the BI system.
2. If errors occur when posting data to the data targets, you can update the data packages from the PSA table to the data targets again after you have eliminated the errors. If the update to the data targets was terminated in the transfer or update rules, you can use the transfer and update rule debug option to search for the error. You can also simulate the update.

3. You can use error handling ('Update' tab page in the InfoPackage) to determine how the system should respond to data with errors.
- **PSA only**

This update type enables you to write all the extracted data packages for a request to the PSA table without updating them to the data targets. A dialog process is started for each data package that writes the data package to the PSA table. The process is available for updating another package when the data in the previous data package has been fully updated to the PSA table.

There are various options available for posting the data from the PSA table to the data targets:

- *Continue update automatically*

If the 'Update to Data Targets Afterwards' indicator is set, a process is triggered after all the data packages for a request have been written to the PSA table that writes the data packages to the data targets one after another.

- *Schedule further update*

You can schedule the further updating of a request using a scheduler. This gives you the option of triggering the further update at another point in time.



Hint: When the request is updated further, a batch process is started. You should therefore make sure that sufficient batch processes are available.

- **Data targets only**

With this update type the data packages are updated directly to the data targets. This means that they are not temporarily stored in the PSA table.



Note: if you load the data using the PSA table, you have the option of creating a **start routine** in the maintenance screen for the transfer rules. This is run for each data package after the data is written to the PSA table and before the transfer rules are executed. With the start routine you can change (for example, add or delete data) or check data packages. It is not possible to use start routines for the 'IDoc' transfer method.

Update rules 3.x

Transformations are needed to update data that specify the transfer of the data from the communication structure for the InfoSource to the data targets (physical InfoProvider). To distinguish them more clearly from the transfer rules, these transformations are described as *update rules*.

As was mentioned in the previous section, you have to create update rules if you supply a data target from an InfoSource with flexible update. You create update rules in the context menu for the respective InfoProvider in the Data Warehousing Workbench. Only Standard InfoCube, 'standard' type DataStore object and characteristic InfoObject (attributes/texts) data targets can be used as DataProviders here.

A data target can be supplied by several InfoSources. You have to maintain a set of update rules for each of these InfoSources that describes how the data is to be written from the communication structure for the InfoSource to the data target. Update rules are therefore data target-specific.

The following graphic provides another summary of the most important statements regarding update rules:

The following graphic shows examples of update rules for a Standard InfoCube. These are discussed in relation to all data target types in the next section.

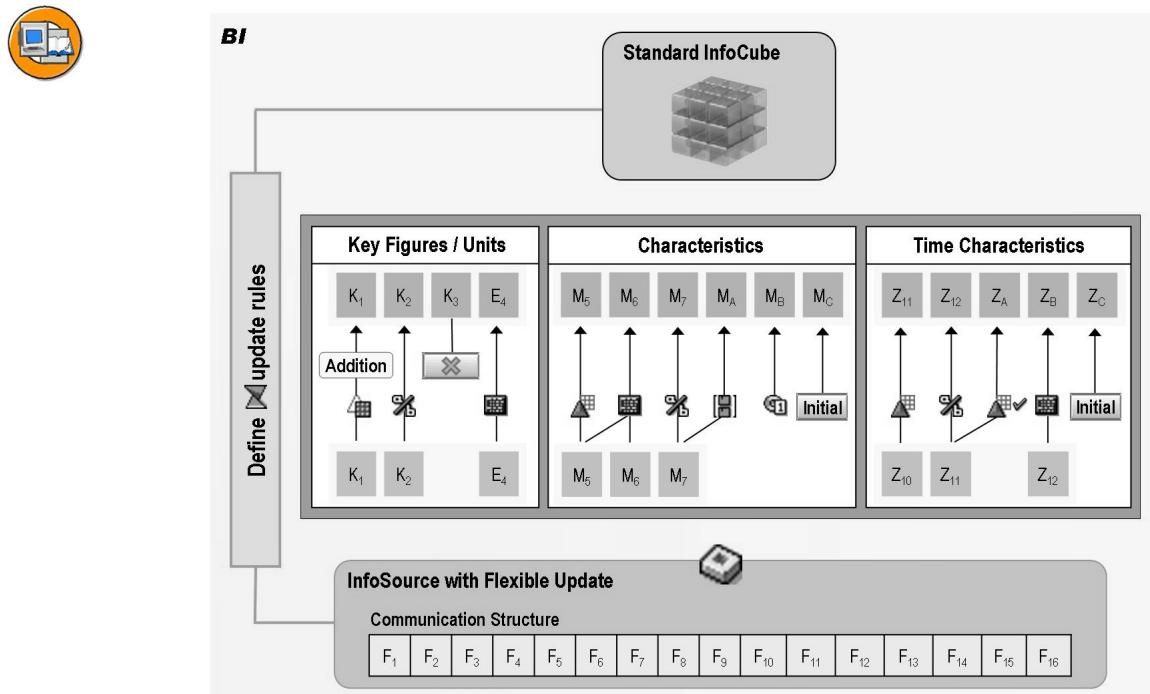


Figure 28: Overview: Update Rules 3.x

When maintaining update rules, you distinguish between the *update method* and the *update type*:

- **Update method**

The update method describes the form in which the values of a characteristic or key figure InfoObject, or a data field/key field, or an attribute/key field are determined or calculated for updating to the data target (Standard InfoCube, or DataStore object, or characteristic InfoCube). The following options are available:

- *Source InfoObject*

(type: Characteristic, key figure, unit, time characteristic)

The InfoObject for the data target is filled directly from the source InfoObject. Time characteristic and unit InfoObjects play a special role in Standard InfoCubes here: If the source InfoObject is a time characteristic, BI provides automatic time conversion for Standard InfoCubes (for example, the conversion of calendar day to calendar week or month). No explicit definition is required for the update of values in a unit InfoObject (currencies/units of measure). Rather, it is inextricably linked to the update of the respective key figure. Consequently, depending on the particular constellation of source currency and target currency, you can use conversion types or ABAP routines to carry out the currency translation, for example.

Another update method is the splitting of key figures/data fields/attribute texts. There are two possible options here:

- *Splitting key figures/data fields/ attribute texts*

1. Copy rules
2. Routine with return table

- *Constant*

Assigning a constant allows you to fill an InfoObject in the data target with a fixed value.

- *Formula*

The value for the InfoObject in the data target is determined with the help of a formula.

- *ABAP routine*

The value for the InfoObject in the data target is determined with the help of an ABAP routine.

- *Reading master data attributes*

A characteristic InfoObject/data field/attribute in the data target is updated by reading data in the master data table of a characteristic InfoObject that is part of the InfoSource and has the characteristic InfoObject/data field/attribute as an attribute. If this relationship does not exist, master data can only be read with the help of an ABAP routine.

- *Initial value*

If the initial value is selected as the Data Sources, the characteristic InfoObject (Standard InfoCube) or the key field (DataStore object/characteristic InfoObject) is updated with the initial value of the respective Data Sources. This does not mean that a value is omitted in the update, but that the values 'SPACE' (with alphanumeric field types) or '0' (with numeric field types) are updated.

The update type (see below) controls whether a key figure, a data field or an attribute/text is updated.

- **Update type**

The update type controls whether and how a key figure, data field or attribute/text is updated in the data target (Standard InfoCube, DataStore object or characteristic InfoObject) with an individual key combination.

- Update types for Standard InfoCubes:

Addition

Maximum

Minimum

No update

Depending on the type of standard aggregation you have specified in maintenance for key figure InfoObjects, the system offers you the choice of 'addition', 'maximum' or 'minimum' for the update type. If you select one of these options, the new key figure values are updated to the Standard InfoCube. With 'addition', for example, the corresponding value from the source key figure is added to the existing values with the same key combination.

If by contrast you select 'no update', no new values for this key figure are updated to the Standard InfoCube.



Hint: 'No update' does not mean that the entire data record is not updated, however. The setting only refers to the respective key figure. All other key figures are updated in the 'normal' way.

- Update types for DataStore objects:

Overwrite

Addition

No update

Depending on the InfoObject type and the DataSource, the system offers you the choice of 'no update', 'addition' and 'overwrite' as update types for the DataStore object data fields. If you choose either of the last two options, new values are updated to the DataStore object.

'Addition' is only possible for data fields belonging to the 'key figure' InfoObject type. This means that addition is not possible for 'characteristic' type data fields. With delta updates you should also bear in mind that the DataSource supports an additive delta (additive image).

In contrast to Standard InfoCubes, you can 'overwrite' existing data when updating to DataStore objects. With 'overwrite', existing values are overwritten by the corresponding value in the source InfoObject with the same key combination. Overwriting enables you to update data at the document and data record levels. You can overwrite all the data fields (characteristics and key figures). Whether or not you are allowed to overwrite a key figure also depends on the delta process for the DataSource. A DataSource could provide an additive delta for a key figure, for example. In this case, overwriting would lead to an incorrect result.

The 'no update' update type behaves in the same way as for Standard InfoCubes.



Hint: The DataStore object has the characteristic InfoObject '0RECORDMODE' (record mode). You can use the record mode to control the delta logic supplied by the application. The characteristic InfoObject '0RECORDMODE' in the DataStore object controls whether the records are added, overwritten or even deleted. It determines how a record is updated in the delta process, the delta process itself being determined by the DataSource. The delta logic is discussed in detail in unit 3 '*Delta Management: Overview*'.

- Update types for characteristic InfoObjects (attributes/texts):

Overwrite

No update

The 'no update' update type behaves in the same way as for Standard InfoCubes/DataStore objects. The 'overwrite' update type behaves in the same way as for DataStore objects.

The following graphic compares the possible aggregation types in relation to the respective data target type.



BI		InfoCube	Data Store Object		Char.-InfoObject
Update Type	Key Figure	Data Field		Attribute / Text	
		Char.	Key Figure		
Addition	X	-	X	-	
Maximum	X	-	X	-	
Minimum	X	-	X	-	
Overwrite	-	X	X	X	
No Update	X	X	X	X	

Figure 29: InfoProvider Aggregation Types



Hint: Similar to the transfer rules, you have the option of creating a **start routine** in update rule maintenance. The start routine is run for each data package and before the update rules are executed.

Migration to SAP NetWeaver 2004s BI Data Transformation Process

You can migrate a DataSource 3.x to a DataSource. If the DataSource 3.x already exists in a data flow for the BI 3.x concept, you use emulation first to model the data flow with transformations and data transfer processes and then test it. Subsequently, you can deactivate the data flow you were using before and delete the metadata objects during migration.



Comparison of two concepts

	SAP NetWeaver 2004 or earlier	SAP NetWeaver 2004s or later
Structure	Transfer Rules & Update Rules	Transformation
Characteristic update	Per key figure	Per transformation group
User interface	Several levels of detail	Aggregation, Source and Target at one glance
Additional Transformation Routines	Start Routine	Start Routine, End Routine, Expert Routine
Unit Conversion	Not implemented	Possible
Hierarchy transformations	Possible	Not yet implemented in SAP NetWeaver 2004s
Return table	Possible	Not yet implemented in SAP NetWeaver 2004s

Figure 30: Comparison of Data Transformation Processes

When you migrate a DataSource 3.x in an original system, the system generates a DataSource with a transport connection. The DataSource 3.x is deleted, along with the 3.x metadata objects mapping and transfer structure , which are dependent on it. If a PSA or InfoPackages already exist for the DataSource 3.x, they are transferred to the migrated DataSource, along with the requests that have already been loaded. After migration, only the information about how data is loaded into the PSA is used in the InfoPackage.

You have the option to export the 3.x objects, DataSource 3.x, mapping and transfer structure during the migration so that these objects can be restored. The collected and serialized objects are stored in a local table (RSDSEXPORT). The migration is now eligible to be transported into the target system

When you import the transport into the target system, in the after import, the system migrates the DataSource 3.x (as long as it is available in the target system) to a local DataSource , without exporting the objects that are to be deleted. The DataSource 3.x, mapping and transfer structure objects are deleted and the related InfoPackages are migrated. The data in the DataSource is transferred to the PSA.

Emulating DataSource 3.x

Emulation before migration is used to model and test the functionality of the (SAP NetWeaver 2004s) data flow with transformations, without changing or deleting the objects of the existing data flow and serves as a first step to migrating DataSources 3.x to DataSources in BI. As a prerequisite, the DataSource 3.x to be migrated must be active along with the transfer structure and transfer rules.

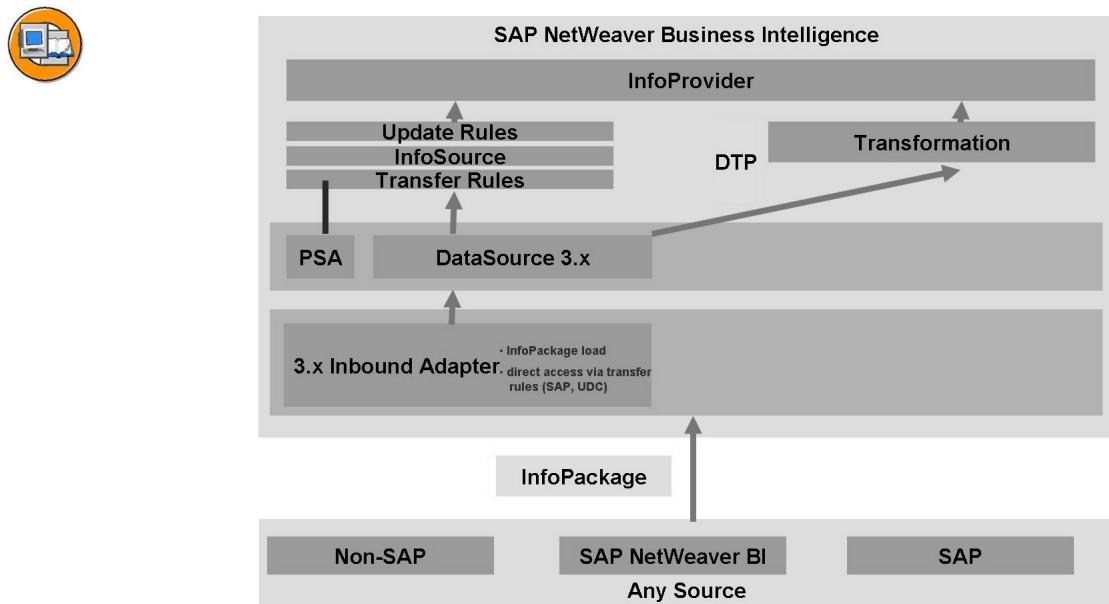


Figure 31: DataSource 3.x Emulation

An emulated DataSource 3.x can be displayed in DataSource maintenance in BI. Changes are not possible in this display. In addition, you can use emulation to create the data flow for a DataSource 3.x with transformations, without having to migrate the existing data flow that is based on the DataSource 3.x.

To display the DataSource 3.x emulated in the DataSource maintenance, mark the DataSource 3.x in the DataSource tree and choose *Display* in the context menu. To create the data flow using transformations, mark the DataSource 3.x in the DataSource tree and choose *Create Transformation* in the context menu. You set the target for the data transfer from the PSA table using the transformation.

Migrating DataSource 3.x

You perform migration in the original system.

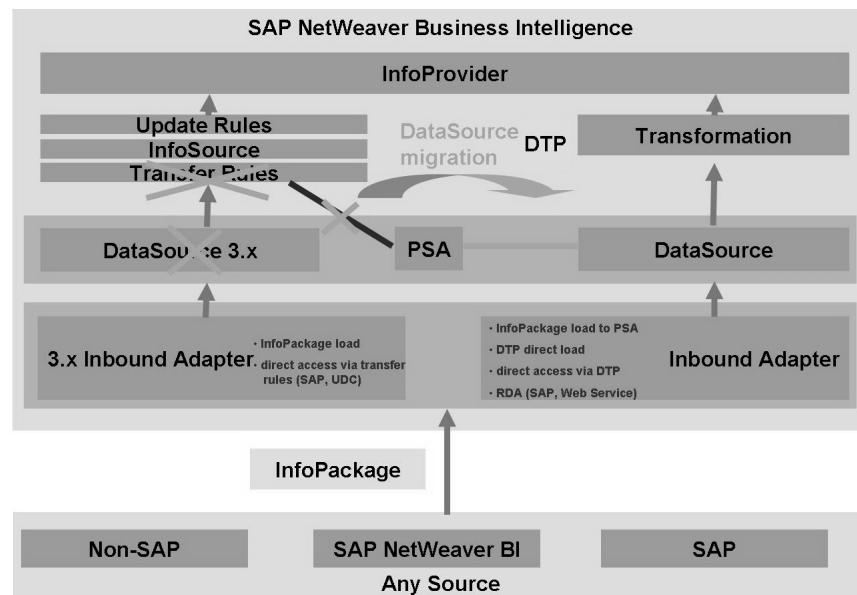


Figure 32: DataSource 3.x Migration

You import the transport for the migration into the target system according to the following steps:

- In the original system (development system), in the Data Warehousing Workbench, choose *Migrate* in the context menu of the DataSource 3.x.
- If you want to restore the DataSource 3.x at a later time, choose *With Export* on the next screen.
- Specify a transport request.
- Transport the migrated DataSource to the target system (quality system, productive system).

Special Features of Source-System-Dependent Migration

If you want to transfer data to BI using a Web service and have already implemented XML DataSources that were generated on the basis of a file DataSource, you cannot perform migration in the way described above. You can make a copy of the generated XML DataSource in a source system of type *Web Service*. When you activate the DataSource, the system generates a function module and a Web service. On your interface, these are different to the 3.x objects. The 3.x objects (DataSource 3.x, mapping, transfer rules) are therefore obsolete and can be deleted manually. If you want to copy an XML DataSource to a Web service source system, check whether you need to make adjustments in the source.

→ **Note:** This restriction is also valid for UDC DataSources which were generated on the basis of a file DataSource.

Restoring DataSource 3.x

If you exported and archived the 3.x metadata objects when you migrated the DataSource 3.x in the original system, you can make a SAP NetWeaver 2004s DataSource 3.x from the DataSource. The system reproduces the DataSource 3.x, mapping, and transfer structure objects with their pre-migration status.

When you restore, the DataSource 3.x, mapping and transfer structure objects that were exported are generated with a transport connection in the original system. DataSource is deleted. The system tries to retain the PSA. This is only possible if a PSA existed for the DataSource 3.x before migration. The InfoPackages for the DataSource are retained in the system. However, in InfoPackage maintenance you have to reselect the targets into which you want to update data.

 **Note:** Only use the restoration function if unexpected problems occur with the SAP NetWeaver 2004s data flow after migration and the problems can only be solved by restoring the data flow used previously.

- In the original system (development system), in the maintenance for the migrated DataSource (transaction RSDS), choose *DataSource → Restore* and specify a transport request.
- If required, delete the dependent transformation and data transfer process objects.
- Transport the restored DataSource 3.x, along with its dependent objects, into the target system.

When you restore, the DataSource 3.x , mapping and transfer structure objects that were exported are generated with a transport connection in the original system. DataSource is deleted. The system tries to retain the PSA. This is only possible if a PSA existed for the DataSource 3.x before migration. This may not be the case if an active transfer structure did not exist for the DataSource 3.x or if the data for the DataSource was loaded using an IDoc. The InfoPackages for the DataSource are retained in the system. Available targets are displayed in the InfoPackages (this also applies to InfoPackages that were created after migration). However, in InfoPackage maintenance you have to reselect the targets into which you want to update data.

When you transport the restored DataSource 3.x into the target system, the DataSource is deleted in the after image. The PSA and InfoPackages are retained. If a transfer structure is transported with the restore process, the system tries to transfer the PSA for this transfer structure. This is not possible if no transfer structure exists when you restore the DataSource 3.x or if IDoc is determined as the transfer method for the DataSource 3.x. The PSA is retained in the target system but is not assigned to a DataSource/DataSource 3.x or to a transfer structure.



Lesson Summary

You should now be able to:

- Describe the data transfer process
- Contrast the new data flow with the previous data flow
- Describe the conversion path to the new data transfer process

Related Information

- [Enter an optional reference using the URL or CrossReference tag to additional information that learner may find useful. Examples include websites or whitepapers. Delete if not used.]

Lesson: Transformation Process

Lesson Overview

This lesson discusses transformation and explains the functions that can be used for transformation during data extraction.



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the transformation process
- List the functions available for transformation of data during extraction

Business Example

Your company would like to set up reporting using sales and purchasing data from your SAP source system in BI. You have previously extracted this data and would like to use the new transformation functions.

Transformation Functions

The transformation process allows you to consolidate, cleanse, and integrate data. You can semantically synchronize data from heterogeneous sources. When you load data from one BI object into a further BI object, the data is passed through a transformation. A transformation converts the fields of the source into the format of the target.

You create a transformation between a source and a target. The BI objects DataSource, InfoSource, DataStore object, InfoCube, InfoObject and InfoSet serve as source objects. The BI objects InfoSource, InfoObject, DataStore object and InfoCube serve as target objects. The following graphic illustrates how the transformation is integrated in the dataflow:

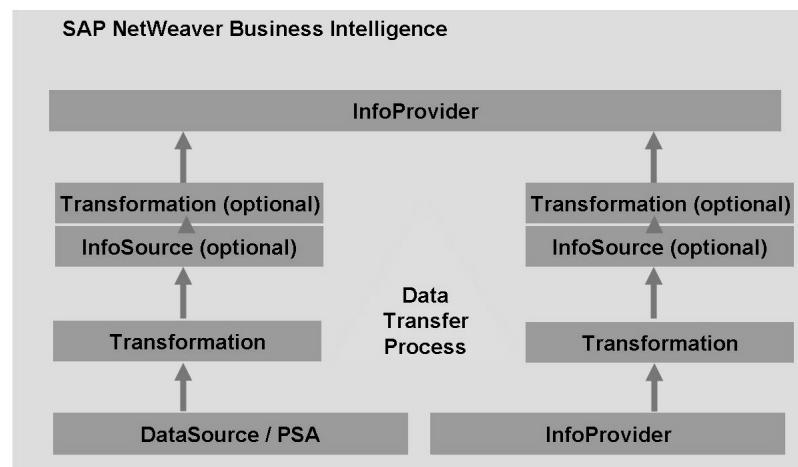


Figure 33: Data Transfer Process

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 to highly complex transformations.

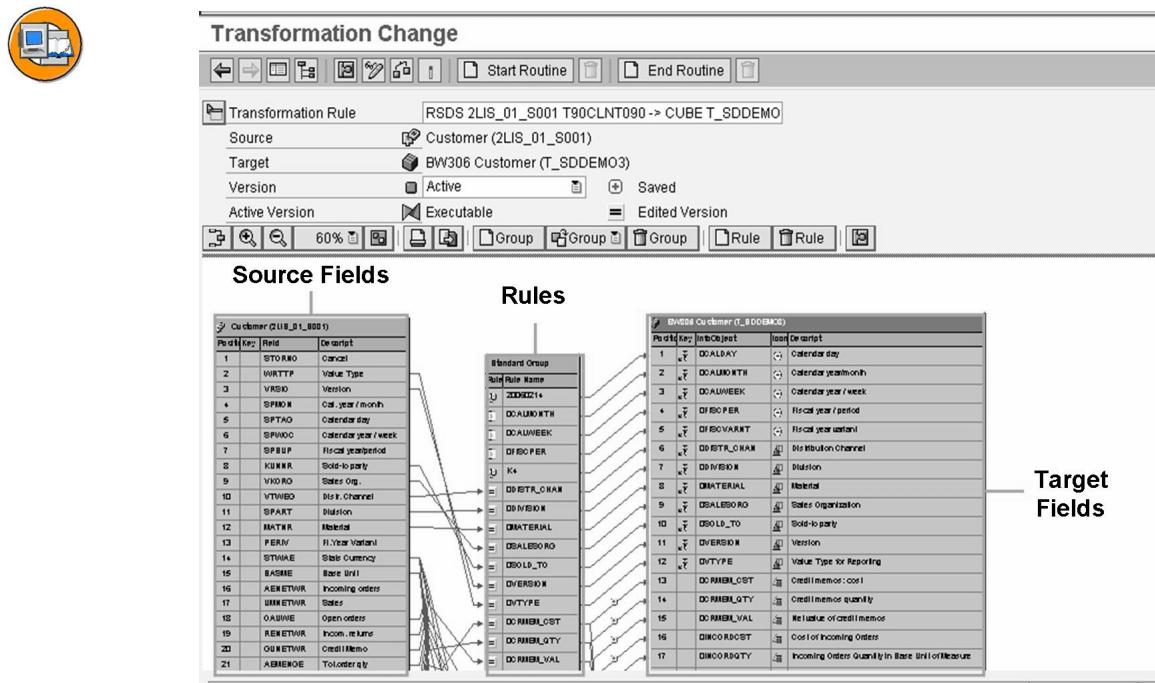


Figure 34: Transformation Rules Overview

- Transformation rules: Transformation rules map any number of source fields to at least one target field. You can use different rule types for this.
- Rule type: A rule type is a specific operation that is applied to the relevant fields using a transformation rule.
- Transformation type: The transformation type determines how data is written into the fields of the target.
- Rule group: 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 you can create different rules for different key figures for a characteristic.

For example, the source contains three date characteristics: order date, delivery date, and 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.

- Routine: You use routines to implement complex transformation rules yourself. Routines are available as a rule type. There are also routine types that you can use to implement additional transformations.

Rule Type

The rule type determines whether and how a characteristic/key figure or a data field/key field is updated to the target. The following options are available:

- 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 of the same type that has a different currency to the target InfoObject, 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 of the same type that has a different unit of measure to the target InfoObject, 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.

- Constants

The field is not filled by the InfoObject but is filled directly with the value specified

- Formula

The InfoObject is updated with a value determined using a formula. Predefined formulas are available in the transformation library which is available in the maintenance for transformation rules (and in the update rules). You can use this in connection with the formula builder.



Caution: Do not use formulas for VirtualProviders because inversion is not allowed for them. Use routines in this case

The transformation library, in collaboration with the formula builder, enables you to easily create formulas, without using ABAP coding. The transformation library has over 70 pre-defined functions, in the following categories:

- Functions for character strings
- Date functions
- Basic functions
- Mathematical functions
- Suitable functions
- Miscellaneous functions
- Self-defined functions

You have the option to implement self-defined functions in the transformation library of the formula builder. You can integrate existing function modules in these self-defined functions. In doing so, you

can also make available special functions to be used frequently that are not contained in the transformation library. The Business Add-In **RSAR_CONNECTOR** is available for this.

- **Reading Master Data**

The InfoObject is updated by reading the master data table of a characteristic that is included in the source with a key and a value and contains the corresponding InfoObject as an attribute. The attributes and their values are read from the key, these are then returned.

For example, the *Financial Management Area* characteristic is in the target, but it is not in the source as a characteristic. However, there is a characteristic (cost center, for example) that has the *Financial Management Area* characteristic as an attribute in the source. You can read the *Financial Management Area* attribute from the master data table and use it to fill the *Financial Management Area* characteristic in the target



Caution: It is not possible to read recursively, that is, to read additional attributes for the attribute of the main characteristic. You have to use routines for this.



Hint: If you have changed master data, you have to execute the change run. By reading the master data, the active version is read. If this is not available, an error is raised

If the attribute is time dependent, you also have to define when it should be read: at the current date (system date), at the beginning or end of a period (defined by a time characteristic of the InfoSource), or at a constant date that you enter directly. System date is used as the default

- **Routine**

The field is filled by the transformation routine you have written. The system offers you a selection option that lets you decide whether the routine is valid for all of the attributes belonging to this characteristic, or only for the attributes displayed. Transformation rules generally only have one return value. If you select Return Table, the corresponding key figure routine no longer has a return value; it has a return table. You can then generate any number of values from a data record.



Hint: 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.

If, for the same characteristic, you generate different rules for different key figures/data fields, a separate data record can be created from a data record of the source for each key figure. With InfoCubes: You can also select Routine with Unit. In the routine, you then also get the return parameter ‘UNIT’. You can store the required unit of the key figure, such as ‘ST’, in the parameter. You can use this option, for example, to convert the unit KG in the source, into tons in the target. If you fill the target key figure from a transformation routine, the currency translation has to be performed using the transformation routine. This means that an automatic calculation is not available

- Time Update

When performing a time update, automatic time conversion and time distribution are available. Direct update: the system automatically performs a time conversion.

Time conversions: You can update source time characteristics to target time characteristics using automatic time conversion. This function is not available for DataStore objects, since time characteristics are treated as normal data fields. The system only presents you with the time characteristics for which an automatic time conversion routine exists.

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 performs time broadcasting in the transformation. Time broadcasting always applies to all key figures.

- Initial: The field is not filled. It remains empty
- Unit of Measure Conversion and Currency Translation: You can convert data records into the unit of measure or currency in the target transformation.

Transformation Type

You use the transformation type to control whether a key figure/data field is updated to the InfoProvider.

For InfoCubes, depending on the aggregation type you specified in key figure maintenance for this key figure, you have the options *Addition* or *Maximum* or *Minimum*. If you choose one of these options, new values are updated to the InfoCube. The aggregation type (addition, minimum & maximum) determines how key figures are updated, if the primary keys are the same. For new values, either the total, the minimum or the maximum for these values is formed. If you choose *no transformation*, the key figures are not updated to the InfoCube. This means that no data records are written to the InfoCube with the first data transfer, or that data records that already exist will remain in place in spite of subsequent transfers

For InfoObjects, only the *Overwrite* option is available. With this option, new values are updated to the InfoObject.

For DataStore Objects, depending on the type of data and the DataSource, you have the options *Addition*, *Minimum*, *Maximum* or *Overwrite*. If you choose one of these options, new values are updated to the DataStore object. For numerical data fields, the system uses characteristic 0RECORDMODE to propose an update type. If only the after-image is delivered, the system proposes *Overwrite*.

 **Note:** More details of the dependency of updating and record images is discussed in the unit, *Delta Management*.

Routines In the Transformation

You use routines to define complex transformation rules. Routines are local ABAP classes that consist of a predefined definition area and an implementation area. The TYPES for the inbound and outbound parameters and the signature of the method are determined in the definition area. The actual routine is created in the implementation area. In the method, ABAP object statements are available. Upon generation, this method is embedded in the local class of the transformation program. Routines are available as rule types; you can define a routine as a transformation rule for a key figure or a characteristic.



Start Routine

- **Use**
 - ◆ Preparation of data before transformation
 - ◆ Package-based, semantic packaging possible (see data transfer process for more details)
- **Example**
 - ◆ Deletion of records that are not required for updating
 - ◆ Performance: Buffering tables into internal tables that can be used for the transformation rules (rather than reading the database tables one by one)

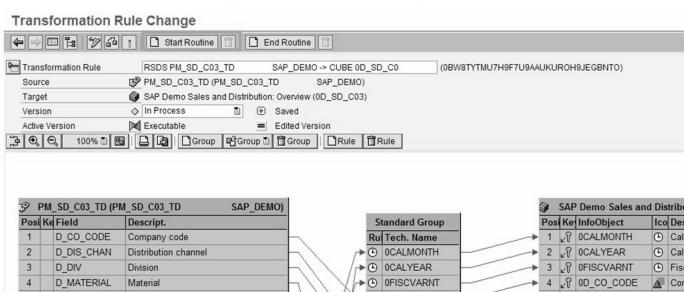


Figure 35: Transformation–Start Routine

Start Routine: The start routine is run for each data package at the start of the transformation. The start routine does not have a return value. 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.



End Routine

■ Use

- ◆ Post-preparation of data after transformation
- ◆ Package-based, semantic packaging possible (see data transfer process for more details)

■ Example

- ◆ Deletion of records after transformation that are not required for updating (e.g. after determining material category for a particular material, every material of type 'refund' is not updated)
- ◆ Validation Checks of records after transformation (e.g. key figure sales value must be bigger than purchase value)

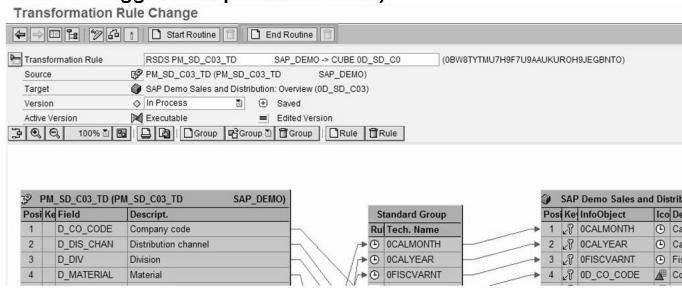


Figure 36: Transformation–End Routine

End Routine: An end routine is a routine with a table in the target structure format as an input parameter and an output parameter. You can use an end routine to execute the postprocessing of data after transformation on a package-by-package basis. For example, you can delete records that are not to be updated, or perform data checks.



Expert Routine

■ Use

- ◆ Transformations that cannot be expressed declaratively for functional or performance reasons

■ Example

- ◆ Performance: reading several database tables can be implemented faster when knowing the application logic (rather than using the generic transformation framework)
- ◆ Pivoting: transpose a wide data record into several smaller records → can be easily implemented using the expert routine

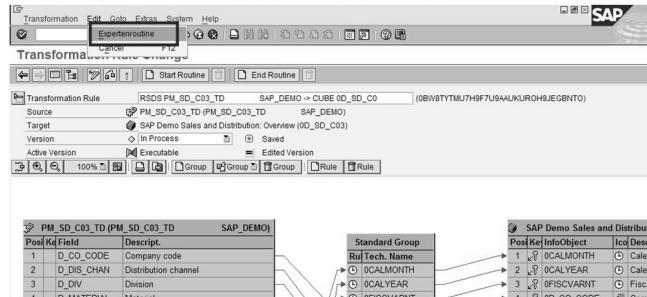


Figure 37: Transformation-Expert Routine

Expert Routine: This type of routine is only intended for use in special cases. You can use the expert routine if there are not sufficient functions to perform a transformation. You can use the expert routine as an interim solution until the necessary functions are available in the standard routine. You can use this to program the transformation yourself without using the available rule types. You must implement the message transfer to the monitor yourself.



Hint: If you have already created transformation rules, the system deletes them once you have created an expert routine.

Currency Translation in the Update

Currency translation in the update allows you to translate data records from the source currency of the InfoSource into a target currency in the data target. This function is only available if key figures with currency fields exist in the source system in various currencies. Currency translation is not available if the currency of the data records to be transferred is the same as the currency of the target key figure in the InfoCube. Currency translation in the update is normally performed using previously defined translation types. It can take place using user-defined subprograms (formulas and routines), if necessary.



Note: For DataStore objects it is currently not possible to execute the currency translation by using predefined translation types. You have to use routines instead.

With the update of key figures, there are two basic different cases:

1. If for every key figure of an InfoCube (target key figure) there is a corresponding key figure in the InfoSource (source key figure), then no currency translation takes place
2. If there is no corresponding source key figure in the InfoSource for the target key figure of the InfoCube, then either a source key figure of the **same type** can be assigned to the target key figure (for example, sales revenue instead of sales quantity revenue) or if there is **no** corresponding source key figure of the **same type**, then you have to fill the key figure for the data target from a **routine**.

If the target key figure has a **fixed currency**, no currency translation is carried out. This means that if translation is required, you have to execute it in the routine

If the target key figure has a **variable currency**, you also have to assign a variable source currency to the routine.

The following table provides an overview of possible combinations with **different currencies in the source and target key figures**.

Source Key Figure currency	Target Key figures currency	Currency Translation (CT)
Fixed	Variable	No CT
Fixed	Fixed	CT
Variable	Fixed	CT
Variable	Variable	CT or assignment possible

Creating a Routine for Currency Translation:

If you want to translate currencies in the update even though the currency translation is not available for one of the above reasons, you can create a routine. Choose *Routine*, set the *Unit Calculation in the Routine* and choose *Create Routine*. In the routine editor you get the additional return parameter UNIT, the value of which is used to determine the target currency.

Quantity Conversion During the Transformation

Quantity conversion during the transformation enables you to convert data records from the source unit of measure into a unit of measure of the target of the transformation. Quantity conversion in the transformation is generally performed using previously defined quantity conversion types

Where required, you can perform quantity conversion with 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 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 so the unit does not have to be entered again in the data record.
2. Key figures with variable unit of measure: A variable unit of measure references an InfoObject.

Transformations can be performed for key figures in two ways:

- Every key figure in an InfoCube (target key figure) has a corresponding key figure in the source (source key figure). Quantity conversion is not performed
- There is no corresponding source key figure in the InfoSource for the target key figure in the InfoCube. In this case:
 - 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.

- 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.

Creating a Routine for Quantity Conversion

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. In transformation rule definition, choose *Routine with Unit*. In the routine editor you get an additional return parameter UNIT and the target unit of measure is determined using the value of this parameter.

Quantity Conversion Types

In the section above, reference was made to the use of quantity conversion types in the case of differing units of measure. A quantity conversion type is a combination of different parameters that establish how the conversion is performed. The parameters that determine the conversion factors are the source and target unit of measure and the option you choose for determining the conversion factors. The following graphic illustrates an overview of the inputs to the quantity conversion type



Inputs to the Conversion Process

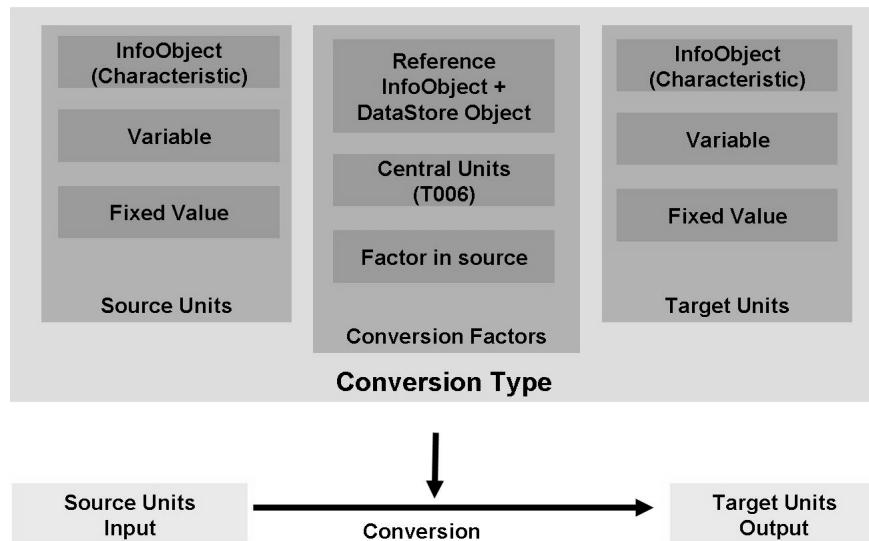


Figure 38: Inputs to the Conversion Process

Conversion Factors

The decisive factor in defining a conversion type is the way in which you want conversion factors to be determined. Entering source and target quantities is optional.

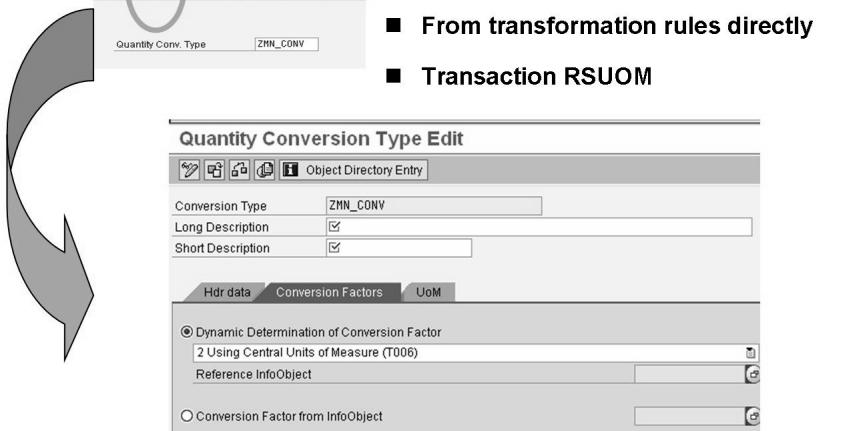
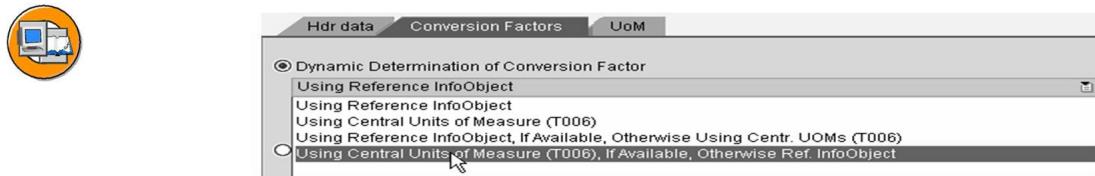


Figure 39: UOM Conversion Factors in General

The specific options for dynamic determination of conversion factors are illustrated in the graphic below:



- Allows for determination to come from either an InfoObject (and the linked UOM DataStore Object) or table T006 or both.
- If “both” then order of operations is set and failing to find it in the initial source, the system will seek the alternate source for final determination

Figure 40: Dynamic Determination of Conversion Factor

The following options are available for conversion factors:

- Using a reference InfoObject: The system tries to determine the conversion factors from the reference InfoObject you have chosen or from the associated quantity DataStore object. If you want to convert 1000 grams into kilograms but the conversion factors are not defined in the quantity DataStore object, the system cannot perform the conversion, even though this is a very simple conversion.
- Using central units of measure: Conversion can only take place if the source unit of measure and target unit of measure belong to the same dimension (for example, meters to kilometers, kilograms to grams, and so on).
- Using reference InfoObject if available, central units of measure if not: The system tries to determine the conversion factors using the quantity DataStore object you have defined. If the system finds conversion factors, it uses these to perform the calculation. If the system cannot determine conversion factors from the quantity DataStore object it tries again using the central units of measure.
- Using central units of measure if available, reference InfoObject if not: The system tries to find the conversion factors in the central units of measure table. If the system finds conversion factors it uses these to perform the conversion. If the system cannot determine conversion factors from the central units of measure it tries to find conversion factors that match the attributes of the data record by looking in the quantity DataStore object.

The settings that you can make in this regard affect performance and the decision must be strictly based on the data set. Consider the following points:

1. If you only want to perform conversions within the same dimension, option 2 is most suitable.
2. If you are performing InfoObject-specific conversions (for example, material-specific conversions) between units that do not belong to the same dimension, option 1 is most suitable.



Note: In both cases above, the system only accesses one database table. That table contains the conversion factors.

3. With option 3 and option 4, the system tries to determine conversion factors at each stage. If conversion factors are not found in the basic table (T006), the system searches again in the quantity DataStore object, or in reverse.
4. The option you choose should depend on how you want to spread the conversion. For example, if the source unit of measure and target unit of measure belong to the same dimension for 80% of the data records that you want to convert, first try to determine factors using the central units of measure (option 4), and accept that the system will have to search in the second table also for the remaining 20%.



Note: The *Conversion Factor from InfoObject* option (as with *Exchange Rate from InfoObject* in currency translation types) is only available when you load data. The key figure you enter here has to exist in the InfoProvider and the attribute this key figure has in the data record is taken as the conversion factor.

Source Unit of Measure

The source unit of measure is the unit of measure that you want to convert. The source unit of measure is determined dynamically from the data record or from a specified InfoObject (characteristic).

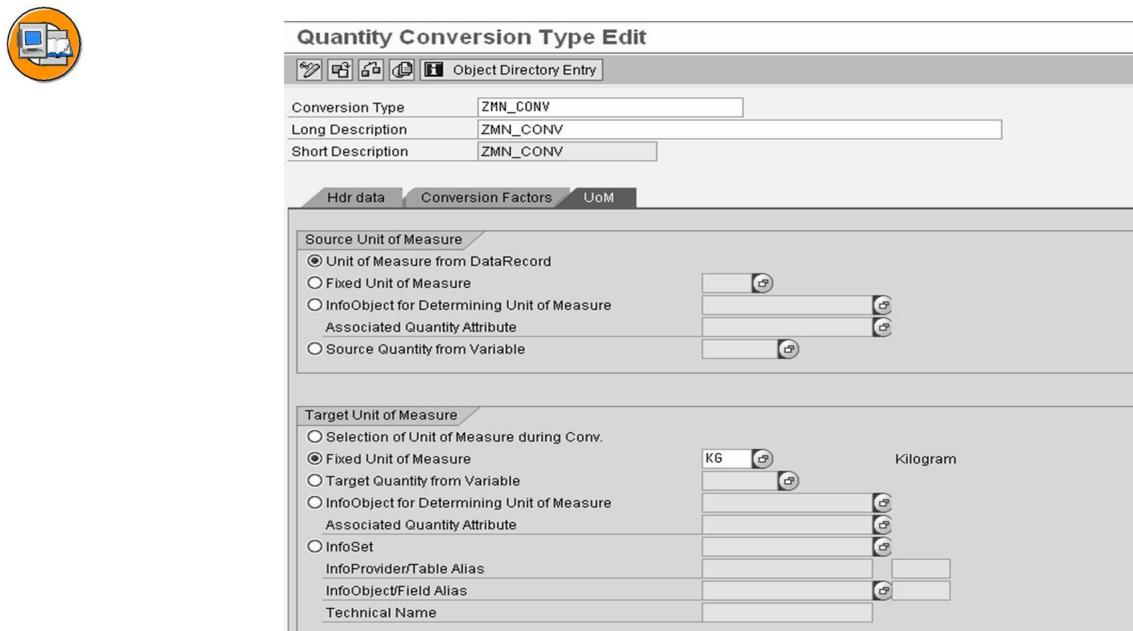


Figure 41: UOM Source and Target Units of Measure

In addition, you can specify a fixed source unit of measure or determine the source unit of measure using a variable. When converting quantities in the Business Explorer, the source unit of measure is always determined from the data record. During the data load process the source unit of measure can be determined either from the data record or using a specified characteristic that bears master data. You reach the maintenance for the unit of measure in *SAP Customizing Implementation Guide* → *SAP NetWeaver* → *General Settings* → *Check Units of Measure*.

Target Unit of Measure

In creating a Unit Conversion Type, entering source and target quantities is optional.

The following graphic depicts using a unit of measure conversion type during Transformation:



Rule Details

Description	0D_00RQTYBM	Open order quantity in base unit (SAP DEMO)				
Target InfoObject	0BASE_UOM	Base Unit of Measure				
Rule Type	Direct Assignment					
Aggregation	Summation					
Unit of measure						
Target Unit	0BASE_UOM	Base Unit of Measure				
Unit of measure	from Conversion					
Conversion Type	PM_CONV1					
Source Fields of Rule:						
InfoObject	Field	Ic.	Long Description	Type	Len...	Conv...
0BASE_UOM	BASE_UOM		Base Unit of Measure	UNIT	6	CUNIT
0D_00RQTYBM	D_00RQTYBM		Open order quantity in base unit	QUAN	9	
Target Fields of Rule:						
InfoObject	Field	Ic.	Long Description	Type	Len...	Conv...
0BASE_UOM	BASE_UOM		Base Unit of Measure	UNIT	6	CUNIT
0D_00RQTYBM	D_00RQTYBM		Open order quantity in base unit	QUAN	9	

Transfer Values | **Transfer** | **Cancel** | **Close**

Condition

- **UOM Conversions** require an '**InfoObject-based**' source, i.e. a **DataSource** cannot be used directly but via an intermediate **InfoSource**

Figure 42: UOM Conversion Type during Transformation

You have the following options for determining the target unit of measure:

- You can enter a fixed target unit of measure in the quantity conversion type (for example, 'UNIT').
- You can specify an InfoObject in the quantity conversion type that is used to determine the target unit of measure during the conversion. Under *InfoObject for Determining Unit of Measure*, all InfoObjects are listed that have at least one attribute of type Unit. You have to select one of these attributes as the corresponding quantity attribute.
- Alternatively, you can specify that the target unit of measure be determined during the conversion.
- Target quantity using InfoSet: This setting covers the same functionality as *InfoObject for Determining Target Quantity*. If the InfoObject that you want to use to determine the target quantity is unique in the InfoSet (it only occurs once in the whole InfoSet), you can enter the InfoObject under *InfoObject for Determining Target Quantity*. You only have to enter the InfoObject in *Target Quantity Using InfoSet* if you want to determine the target quantity using an InfoObject but that occurs more than once in the InfoSet

Transferring Global Table Entries for Units of Measure from SAP Systems

You use this function to transfer all the tables relevant for unit conversion from other SAP systems connected to the BI system. Specifically, this includes the following tables: T006, T006A, T006B, T006C, T006D, T006I, T006J and T006T

In the Data Warehousing Workbench under *Modeling*, choose the source system tree. In the context menu of your *SAP Source System*, choose *Transfer Global Settings*. The *Transfer Global Settings: Selection* screen appears. Under *Transfer Global Table Contents*, select the *Units of Measure* field.



Creating Variables for Quantity Conversion

Use

When you create a quantity conversion type you can use a variable for the source and target unit of measure.

Prerequisites

You have assigned the technical Content InfoObject *Unit* (0UNIT) to an InfoArea so that this InfoObject can be used as an InfoProvider.

Procedure

1. Open the Query Designer.
2. Create a new query for the *Unit* (0UNIT) InfoObject.
3. In the context menu of the InfoObject, choose *New Variable*. The variables editor appears.
4. Enter a description for the variable. If necessary, change the automatically generated suggestion for the technical name of the variable.
5. Choose the required processing type in the *Processing by* field. If you choose *User Entry/Default Value*, the system requests that you enter the unit of measure in a dialog box in the query.
6. On the *Currencies and Units* tab page, select *Quantity* as the dimension.
7. Choose *OK*. The variable is saved with the settings you have made and the variables editor closes. Leave the Query Designer.

Result

The variable can be used in the quantity conversion type.



Creating Quantity Conversion Types

Prerequisites

If you want to use variables in the quantity conversion type, you have to create them beforehand

Procedure

1. On the SAP Easy Access screen for SAP NetWeaver Business Intelligence, choose *SAP Menu → Modeling → Object Maintenance → Unit Conversion Types*.
2. Enter a technical name for the conversion type. The name must be between three and ten characters long and begin with a letter. Choose *Create*. The *Edit Quantity Conversion Type* screen appears, where you enter a description.
3. Choose *Dynamic Determination of Conversion Factor or Conversion Factor from InfoObject*. If you choose *Dynamic Determination of Conversion Factor* you have to enter further information in the dropdown box.
4. Under *Source Unit of Measure*, choose whether the source unit of measure is:
 - determined using the data record, or
 - has a fixed unit of measure, or
 - is determined using an InfoObject, or
 - is determined using a variable.
5. Under *Target Unit of Measure*, choose whether the target unit of measure is:
 - is selected during the conversion, or
 - has a fixed unit of measure, or
 - is determined using a variable, or
 - is determined using an InfoObject.
6. Save your entries.



Caution: Note that with InfoSets, the InfoObject can occur several times within the InfoSet and therefore may not be unique. In this case you have to specify the unique field alias name of the InfoObject in the InfoSet. The field alias name consists of <InfoSet name> <three underscores> <field alias>. You can display the field alias in the InfoSet Builder using *Switch Technical Name On/Off*.

Result

The quantity conversion type is available for converting quantities in the update of InfoCubes and with data analysis in the Business Explorer.

Exercise 2: Data Acquisition Transformation in BI

Exercise Objectives

After completing this exercise, you will be able to:

- To gain an overview of the transformation features offered for data acquisition in BI

Business Example

Your company will be extracting data from a variety of source systems into BI. Not every source system presents all the data necessary. In order to effectively use data acquisition in BI, you want to explore the various options for the transformation of data during the extraction process including the new unit of measure conversion.

Task 1:

Create an InfoSource by using an existing DataSource as a template.

1. Go to the Data Warehousing Workbench for Modeling and find the Source Systems tree. From source system **I_EXTERN** (Folder File), go to DataSource **T_DS_TRANS** found under Global Training Objects within **BW Training**. Display the field attributes of this DataSource, but do not change anything.
2. By using DataSource **T_DS_TRANS** of source system **I_EXTERN** as a template, create a new InfoSource **IS_TRANS##** under Application Component **ZT_BW350_GR##**.

<i>InfoSource</i>	IS_TRANS##
<i>Long Description</i>	Transformation Sales Details ##
<i>Application Component</i>	ZT_BW350_GR##
<i>Copy From Object Type</i>	DataSource
<i>DataSource</i>	T_DS_TRANS
<i>Source System</i>	I_EXTERN

Continued on next page

Task 2:

Create a new Transformation between the DataSource **T_DS_TRANS** and your InfoSource **IS_TRANS##**. Create a new InfoCube and a new Transformation between your InfoSource and your InfoCube.

1. Create a new transformation for your InfoSource **IS_TRANS##** using as the source the DataSource **T_DS_TRANS** based on Source System **I_EXTERN**. Accept the Transformation proposal without any changes.
2. Create a new InfoCube **TRANS##** under InfoArea **T_BW350_GR##** (copied from existing InfoCube **T_TRANS**). Activate the InfoCube.

<i>InfoCube</i>	TRANS##
<i>Description</i>	Sales Details ##
<i>Copy From</i>	T_TRANS
<i>InfoProvider Type</i>	Standard InfoCube
<i>Real Time</i>	No (blank)

3. Create a new transformation for your InfoCube **TRANS##** using as the source the InfoSource **IS_TRANS##**.

Task 3:

Change your Transformation between the InfoSource and the InfoCube to provide the necessary information.

1. Make a change within the Transformation to link *OPOST_DATE* of the InfoSource as a source to the time characteristics *OCALDAY*, *OCALMONTH*, *OCALYEAR* of the InfoCube as target fields. During data upload an **Automatic Time Conversion** will be used to derive the values for the time characteristics.
2. Make a change within the Transformation to link *OCUSTOMER* of the InfoSource as a source to *OCOUNTRY* of the InfoCube as a target, using rule type **Read Master Data**.
3. Make a change within the transformation to set a constant value **010** for *OVTYPE* of the InfoCube.
4. Make a change within the Transformation to do define a **formula transformation** for *OVERSION* to just assign Version ‘000’ whenever the source does not deliver any value here.

Formula String: **IF (VERSION <> '', VERSION, '000')**

Continued on next page

5. Make a change within the Transformation to change **unit transformation** settings to *From conversion* and enter conversion type **T_BW350_Q1** for *OBASE_QTY*. Examine the details behind the conversion type.
6. Make a change within the Transformation to change **currency translation** settings to *From conversion* and enter conversion type **T_BW350_C1** for *ONET_VAL_S*. Examine the details behind the conversion type.
7. Check the overview of your transformation, and then activate it.

Solution 2: Data Acquisition Transformation in BI

Task 1:

Create an InfoSource by using an existing DataSource as a template.

1. Go to the Data Warehousing Workbench for Modeling and find the Source Systems tree. From source system **I_EXTERN** (Folder File), go to DataSource **T_DS_TRANS** found under Global Training Objects within **BW Training**. Display the field attributes of this DataSource, but do not change anything.
 - a) From the SAP Easy Access menu select *Modeling -> Data Warehousing Workbench: Modeling*. If you get a pop-up message about documentation select the check box to not show the question again and press *Yes*.
 - b) Click on *Source Systems* in the Navigator on the left-hand side.
 - c) Click on the *Hide/Show Empty Folders* icon so that all possible source systems groupings are visible.
 - d) Expand *File* and highlight the Source System **I_EXTERN**.
 - e) On right-hand mouse context menu choose *Display DataSource Tree*. The list of DataSources should appear.
- Expand the node *BW Training*, then expand the node *Global Training Objects*.
- f) Highlight the DataSource **T_DS_TRANS**. On right-hand mouse context menu choose *Display*. Consider the field attributes found under the *Fields* tab.
2. By using DataSource **T_DS_TRANS** of source system **I_EXTERN** as a template, create a new InfoSource **IS_TRANS##** under Application Component **ZT_BW350_GR##**.

<i>InfoSource</i>	IS_TRANS##
<i>Long Description</i>	Transformation Sales Details ##
<i>Application Component</i>	ZT_BW350_GR##

Continued on next page

<i>Copy From Object Type</i>	DataSource
<i>DataSource</i>	T_DS_TRANS
<i>Source System</i>	I_EXTERN

- Click on InfoSources in the Navigator. Expand the node **BW350 BI Data Acquisition**, then highlight the application component **ZT_BW350_GR##**. On right-hand mouse context menu choose *Create InfoSource*.
- Enter the above values:
Press the *Transfer (Enter)*  icon.
- Press the *Activate*  icon to activate your InfoSource.

Task 2:

Create a new Transformation between the DataSource **T_DS_TRANS** and your InfoSource **IS_TRANS##**. Create a new InfoCube and a new Transformation between your InfoSource and your InfoCube.

- Create a new transformation for your InfoSource **IS_TRANS##** using as the source the DataSource **T_DS_TRANS** based on Source System **I_EXTERN**. Accept the Transformation proposal without any changes.
 - Find your new InfoSource **IS_TRANS##** in the tree structure under Application Component **ZT_BW350_GR##**. If not, press the *Refresh subtree*  icon.
 - Highlight your new InfoSource **IS_TRANS##** and on right-hand mouse context menu choose *Create Transformation*.
 - In the pop-up window specify the following values:

<i>Source of the Transformation Object Type</i>	DataSource
<i>Source System</i>	I_EXTERN
<i>DataSource</i>	T_DS_TRANS

Press *Create Transformation (Enter)* .

- Accept the Transformation proposal without doing any changes and press the *Activate*  icon.
- Create a new InfoCube **TRANS##** under InfoArea **T_BW350_GR##** (copied from existing InfoCube **T_TRANS**). Activate the InfoCube.

Continued on next page

<i>InfoCube</i>	TRANS_##
<i>Description</i>	Sales Details ##
<i>Copy From</i>	T_TRANS
<i>InfoProvider Type</i>	Standard InfoCube
<i>Real Time</i>	No (blank)

- a) Select *InfoProvider* from the Navigator.
 Highlight the InfoArea **T_BW350_GR##**.
 On right-hand mouse context menu choose *Create InfoCube*.
- b) Specify the values above.
 Choose *Create (F5)* .
- c) You are in the screen *Edit InfoCube*. Press icon *Activate* .
3. Create a new transformation for your InfoCube **TRANS_##** using as the source the InfoSource **IS_TRANS##**.
- a) You should find your new InfoCube **TRANS_##** in the tree structure under InfoArea **T_BW350_GR##**. If not, press the *Refresh subtree*  icon.
- b) Highlight your new InfoCube **TRANS_##** and on right-hand mouse context menu choose *Create Transformation*.
- c) In the pop-up window specify the following values:

<i>Source of the Transformation Object Type</i>	InfoSource
<i>InfoSource</i>	IS_TRANS_##

Press *Create Transformation (Enter)* .

Continued on next page

Task 3:

Change your Transformation between the InfoSource and the InfoCube to provide the necessary information.

1. Make a change within the Transformation to link *OPOST_DATE* of the InfoSource as a source to the time characteristics *0CALDAY*, *0CALMONTH*, *0CALYEAR* of the InfoCube as target fields. During data upload an **Automatic Time Conversion** will be used to derive the values for the time characteristics.
 - a) Select *Hide/Show Navigator*  so that your InfoSource screen is in fullscreen mode.
 - b) Within the *Transformation Create* screen highlight the line for *OPOST_DATE*. Holding down the cursor, draw an arrow from that line to the *0CALDAY* line in the Standard Group box.

Repeat these steps for the remaining time characteristics *0CALMONTH* and *0CALYEAR*.
2. Make a change within the Transformation to link *0CUSTOMER* of the InfoSource as a source to *0COUNTRY* of the InfoCube as a target, using rule type **Read Master Data**.
 - a) Within the *Transformation Create* screen highlight the line for *0CUSTOMER*. Holding down the cursor, draw an arrow from that line to the *0COUNTRY* line in the Standard Group box.

Note that an = symbol now appears in the left-most box of the Standard Group.

 - b) Double-click on this line. A pop-up box *Rule Details* appears.

Specify Rule Type: **Read Master Data** and specify *0CUSTOMER* as source InfoObject.
 - c) Press pushbutton *Transfer Values*.

Note that now the = symbol on *0COUNTRY* has changed to the *Attributes/Master Data* symbol in the left-most box of the Standard Group.

Continued on next page

3. Make a change within the transformation to set a constant value **010** for *0VTYPE* of the InfoCube.
 - a) Within the Standard Group box find the line for *0VTYPE*. Note that no rule is defined.
Double-click on the line and the pop-up box *Rule Details* appears.
 - b) Specify the *Rule Type*: **Constant** and insert constant value: **010**.
 - c) Press pushbutton *Transfer Values*.
Note that now the symbol on *0VTYPE* has changed to the “Constant” symbol in the left-most box of the Standard Group.
4. Make a change within the Transformation to do define a **formula transformation** for *0VERSION* to just assign Version ‘000’ whenever the source does not deliver any value here.

Formula String: **IF (VERSION <> ' ', VERSION, '000')**

- a) Within the Standard Group box, find the line for *0VERSION*. Double-click on the line.
The pop-up box *Rule Details* appears
- b) Specify **Formula** as the *Rule Type* and press the *Change Rule*  icon.
 **Note:** Note: You need to be in Expert mode to key in the formula directly. Press the *Expert mode* pushbutton (spanner icon) to do this.
- c) You want a formula transformation to just assign *Version 000* whenever the source does not deliver any value here. Type in the formula given above exactly in this format or use the offered fields and functions below the formula box. You can select from these as necessary, each with a double-click.
- d) Press the *Check*  pushbutton to check your entries.

When correct, press the *Back*  button and press pushbutton *Transfer Values*.

Continued on next page

5. Make a change within the Transformation to change **unit transformation** settings to *From conversion* and enter conversion type **T_BW350_Q1** for *0BASE_QTY*. Examine the details behind the conversion type
 - a) Highlight the line for *0QUANTITY*. Holding down the cursor, draw an arrow from that line to the *0BASE_QTY* line in the Standard Group box.
 - b) Within the Standard Group box find the line for *0BASE_QTY*. Double-click on the line.

The pop-up box *Rule Details* appears.
 - c) Under *Unit of measure* change Unit of Measure setting using the selector to **From conversion** and specify *Conversion type* **T_BW350_Q1**.
 - d) Press pushbutton *Maintain Conversion Type* (pencil icon) to examine the conversion type.

When the screen *Quantity Conversion Type* appears, press pushbutton *Display (F7)*. Examine the settings under each of the tabs *Hdr data*, *Conversion factors* and *UoM*. **Do not change anything!**
6. Make a change within the Transformation to change **currency translation** settings to *From conversion* and enter conversion type **T_BW350_C1** for *0NET_VAL_S*. Examine the details behind the conversion type.
 - a) Highlight the line for *0NET_VALUE*. Holding down the cursor, draw an arrow from that line to the *0NET_VAL_S* line in the Standard Group box.
 - b) Within the Standard Group box find the line for *0NET_VAL_S*. The pop-up box *Rule Details* appears. Double-click on the line.
 - c) Under *Currency* change Currency setting using the selector to *From Conversion* and specify Conversion type **T_BW350_C1**.
 - d) Press pushbutton *Maintain Conversion Type* (pencil icon) to examine the conversion type. Examine the settings under each of the tabs *Properties*, *Exchange Rate*, *Currncy* and *Time Ref*. **Do not change anything!**
 - e) Keep pressing *Back* until you return to the *Rule Details* pop-up window.
 - f) Press pushbutton *Transfer Values*.

Continued on next page

7. Check the overview of your transformation, and then activate it.
 - a) Examine the content of the screen *Transformation Create*.
 - b) Press the *Activate*  icon.
 - c) Keep pressing *Back*  until you return to the Data Warehousing Workbench.



Lesson Summary

You should now be able to:

- Describe the transformation process
- List the functions available for transformation of data during extraction

Lesson: Direct Access to Source System Data

Lesson Overview

This lesson covers how to set up direct access to source system transaction and master data to allow reporting without data extraction.



Lesson Objectives

After completing this lesson, you will be able to:

- Describe how to access data in BI source systems directly

Business Example

Your company would like to set up reporting using sales and purchasing data from your BI source systems without direct extraction. You have created the relevant DataSources in BI to enable this direct reporting.

Direct Access Without Extraction

Direct access to data in BI source systems without extraction is available using VirtualProviders. A VirtualProvider is an InfoProvider with transaction data that is not stored in the object itself, but which is read directly in reporting.



Direct Access Scenarios - Affected Layer Components

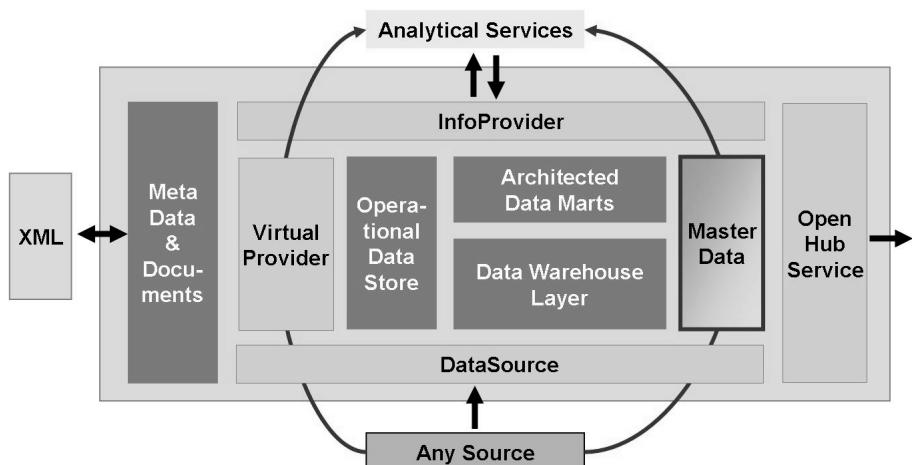


Figure 43: Direct Access without Extraction

The relevant data can be from the BI system, or other SAP or non-SAP systems.

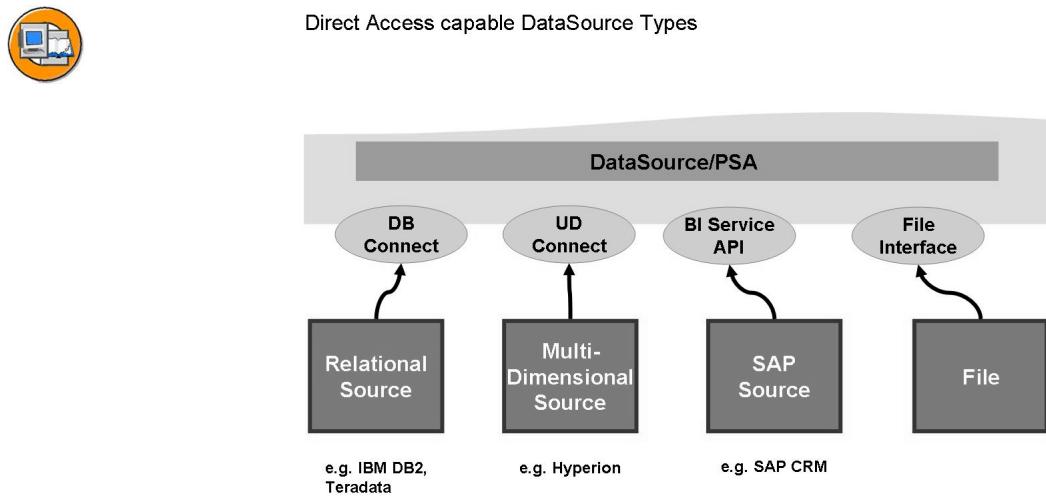


Figure 44: Source Systems Direct Access-capable

→ **Note:** VirtualProviders only allow read access to data.

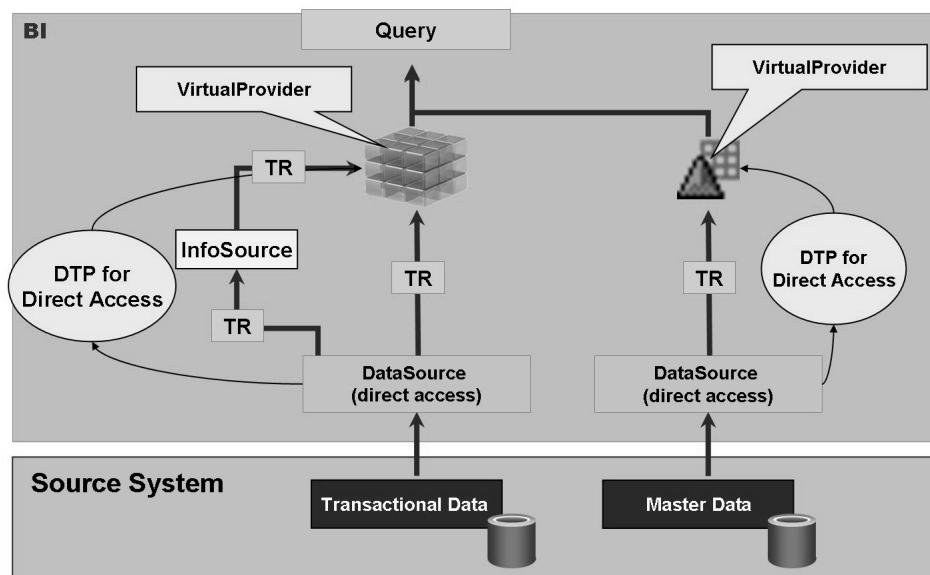
Various VirtualProviders are available depending upon their use in these different scenarios:

- VirtualProviders based on Data Transfer Processes
- VirtualProviders with BAPIs
- VirtualProviders with function modules

The following graphic illustrates that queries based on VirtualProviders provide access to data without extraction and storage:



Direct Access for Transactional and Master Data

**Figure 45: Direct Access by VirtualProvider**

VirtualProviders based on Data Transfer Processes

This type of VirtualProvider is defined based on a DataSource or an InfoProvider and takes on its characteristics and key figures. Unlike other VirtualProviders, you do not need to program interfaces in the source system. You use the same extractors to select data in the source system as you use to replicate data into the BI system. When you execute a query, every navigation step sends a request to the extractors in the assigned source systems. The selection of characteristics together with the selection criteria for these characteristics is transformed, according to the transformation rules, to fields of the transfer structure. They are passed on to the extractor in this form. The delivered data records pass through the transfer rules in the BI system and are then filtered into the query again.



Note: Since master data and hierarchies are not read directly by the source system, they need to be available in the BI system before you execute a query.

If you have specified a constant in the transformation rules, the data is transferred only if this constant can be fulfilled. With more complex transformations such as routines or formulas, the selections cannot be transferred. It takes longer to read the data in the source system because the amount of data is not limited. To prevent this you can create an inversion routine for every transfer routine. Inversion is not possible with formulas, which is why it is recommended that you use routines instead of formulas if you want direct access to data.

Use this VirtualProvider if:

- You need very up-to-date data from an SAP source system
- You only access a small amount of data from time to time
- Only a few users execute queries simultaneously on the database

Do **not** use this VirtualProvider if:

- You request a large amount of data in the first query navigation step, and no appropriate aggregates are available in the source system
- A lot of users execute queries simultaneously
- You frequently access the same data

To create a VirtualProvider based on the Data Transfer Process:

1. In the Data Warehousing Workbench under *Modeling*, choose the InfoProvider tree and in the context menu, choose *Create VirtualProvider*.
2. As the type, select *VirtualProvider based on data transfer process for direct access*.

A VirtualProvider based on a data transfer process with direct access can be linked to an SAP source system using an InfoSource 3.x. Since the transfer rules that are used can also be used for the normal load process to other InfoProviders, you must select the source systems that are to be used for direct access if you use a VirtualProvider that is linked to an InfoSource 3.x.

The *Unique Source System Assignment* indicator controls whether this source system assignment needs to be unique. If the indicator is set, you can select one source system at most in the assignment dialog. If the indicator is not set, more than one source system can be selected. In this case the VirtualProvider acts like a MultiProvider. If the indicator is not set, the characteristic 0LOGSYS is automatically added to the VirtualProvider when it is created. In the query, this characteristic allows you to select the source system dynamically.

3. Define the VirtualProvider by copying the necessary InfoObjects. Activate the VirtualProvider.
4. In the context menu for the VirtualProvider, select *Create Transformation*. Define the transfer rules and activate them
5. In the context menu on the VirtualProvider, select *Create Data Transfer Process*. The default value for the DTP type is *DTP for Direct Access*. Select the source for the VirtualProvider. Activate the data transfer process.
6. Activate direct access. From the context menu of the VirtualProvider, select *Active Direct Access*. In the dialog box that appears, choose one or more data transfer processes and select *Save Assignments*.

VirtualProvider with BAPIs

Using this VirtualProvider, you can carry out reporting on data in external systems without having to physically store transaction data in the BI system. You can, for example, include an external system from market data providers using a VirtualProvider.

This VirtualProvider allows you to connect non-SAP systems, in particular structures that are not relational (hierarchical databases). You can use any read tool that supports the interface for a non-SAP system. The non-SAP system transfers the requested data to the OLAP processor using the BAPI.

When you use this VirtualProvider for reporting, the data manager calls the VirtualProvider BAPI, instead of an InfoProvider filled with data, and transfers the parameters. To use a VirtualProvider with a BAPI for reporting, you have to perform the following steps:

1. In the BI system, create a source system for the external system that you want to use.
2. Define the required InfoObjects
3. Load the master data.
4. Define the VirtualProvider.
5. Define the queries based on the VirtualProvider.

VirtualProvider with Function Modules

You use this VirtualProvider with a user-defined function module, if you want to display data from non-BI data sources in BI without having to copy the dataset into the BI structures. The data can be either local or remote. You can also use your own calculations to change the data before it is passed to the OLAP processor.



Note: This function is used primarily in the SAP Strategic Enterprise Management (SEM) application.

In comparison to other VirtualProviders, this VirtualProvider is more generic. As such it offers more freedom, but also requires more implementation effort.

You specify the type of the VirtualProvider when you create it. If you choose *Based on Function Module* as the type for your VirtualProvider, an extra *Detail* pushbutton appears on the interface. This pushbutton opens an additional dialog box, in which you define the services.

Enter the name of a function module that you want to use as the data source for the VirtualProvider. There are different default variants for the interface of this function module.

You can choose options to support the selection conditions. You do this by selecting the *Convert restrictions* option. These conversions change the transfer table in the user-defined function module only. The result of the query is not changed, because the restrictions that are not processed by the function module are checked later on in the OLAP processor. The options are:

No support: If this option is selected, no restrictions are passed to the function module.

Global selection conditions only: If this option is selected, only global restrictions (FEMS = 0) are passed to the function module. Other restrictions (FEMS > 0) that are created, for example, by setting restrictions on columns in queries, are deleted.

Hierarchies: If this option is switched on, the relevant InfoProvider supports hierarchy restrictions. This is only possible if the InfoProvider also supports SIDs

Do not transform selection conditions: If this option is switched on, all selection conditions are passed on to the function module, without being converted first.

Pack RFC: This option packs the parameter tables in BAPI format before the function module is called, and unpacks the data table that is returned by the function module after the call is complete. Since this option only makes sense in connection with a Remote Function Call, you must define a logical system that is used to determine the target system for the Remote Function Call when you select this option.

SID support: If the data source of the function module can handle SIDs, you should select this option. If this is not possible, the characteristic values are read from the data source and the Data Manager determines the SIDs dynamically.

With navigation attributes: If this option is selected, navigation attributes and restrictions applied to navigation attributes are passed to the function module. If this option is not selected, the navigation attributes are read in the Data Manager once the user-defined function module has been executed

Internal format (key figures): In SAP systems a separate format is often used to display currency key figures. The value in this internal format is different from the correct value in that the decimal places are shifted. You use the currency tables to determine the correct value for this internal representation . If this option is selected, this conversion is used in the calculation in the OLAP processor.

Data quality settings: Sorted data is delivered or “Exact” data is delivered.

Exercise 3: Direct Access for Master Data

Exercise Objectives

After completing this exercise, you will be able to:

- To demonstrate the functionality of direct access for master data

Business Example

Your company has decided to access transaction data from an external system for reporting without storing it persistently in BI using direct access for transaction data. Now you would like to enhance this transactional data with a text field from the external system master data. You will do this using the functionality of direct access for master data.

Task:

To access to text for master data directly you want to build a DataSource with master data texts.

- Create an InfoObject Catalog **T_BW350C##** in your InfoArea **T_BW350_GR##**.

<i>Inf.ObjCat.</i>	T_BW350C##
<i>Description</i>	GR## Characteristics
<i>InfoObject type</i>	Char.

- Create a Remote Characteristic **CUST##** in an InfoObject Catalog **T_BW350C##**.

<i>Char.</i>	CUST##
<i>Description</i>	GR## Direct Customer
<i>Template</i>	OCUSTOMER

On the *Attributes* tab **deselect** all Navigational Attributes.

On the *Master data/texts tab* change the *Master Data Access* to **Remote** from the selection list.

Save your InfoObject CUST## by choosing the *Save* icon.

Activate your InfoObject CUST## by choosing the *Activate* icon.

Continued on next page

3. In order to make your Characteristic available for direct access, your InfoObject **CUST##** must be inserted into the InfoProvider tree as an InfoProvider.
4. Create a Direct Access **DataSource DB_DCUST##** for Source System **ZMSS_00** (DB Connect Source System) under your Application Component **ZT_BW350_GR##**.

<i>DataSource</i>	DB_DCUST##
<i>Data Type DataSource</i>	Master Data Text

On the *General Info.* tab enter **GR## Direct Access** for the short, medium and long descriptions.

On the *Extraction* tab, select **Allowed** from the list in the field *Direct Access*.

This DataSource will access the requested master data texts from a DB Connect Source (Data Base View) which is connected to the BI system. In the field *Table/View* field enter the data base view name **RUDICUSTOMERS**.

5. Create a Transformation between your DataSource **DB_DCUST##** and your InfoProvider **TEXTS CUST##** and map the following fields.

CUSTID	CUST##
CUSTNAME	0TXTMD

6. Create a direct access Data Transfer Process for your InfoProvider **TEXTS CUST##**.
7. Check the results of the direct access by using transaction LISTCUBE for direct access of master data texts.

Result

Congratulations now you see the contents of the database view by direct access.

Solution 3: Direct Access for Master Data

Task:

To access to text for master data directly you want to build a DataSource with master data texts.

1. Create an InfoObject Catalog **T_BW350C##** in your InfoArea **T_BW350_GR##**.

<i>Inf.ObjCat.</i>	T_BW350C##
<i>Description</i>	GR## Characteristics
<i>InfoObject type</i>	Char.

- a) Use the menu *Data Warehousing Workbench: Modeling* → *InfoObjects* → *InfoArea: BW Training* → *BW Customer Training* → *BW350 BI Data Acquisition* → *BW350 Group ## (T_BW350_GR##)* and from the context menu of your InfoArea choose *Create InfoObject Catalog* and specify name and description as given below.

<i>Inf.ObjCat.</i>	T_BW350C##
<i>Description</i>	GR## Characteristics
<i>InfoObject type</i>	Char.

- b) Activate your InfoObject Catalog by choosing the *Activate*  icon and then choose the Back  icon to return to the InfoArea tree.

2. Create a Remote Characteristic **CUST##** in an InfoObject Catalog **T_BW350C##**.

<i>Char.</i>	CUST##
<i>Description</i>	GR## Direct Customer
<i>Template</i>	0CUSTOMER

On the *Attributes* tab **deselect** all Navigational Attributes.

On the *Master data/texts* tab change the *Master Data Access* to **Remote** from the selection list.

Save your InfoObject CUST## by choosing the *Save* icon.

Continued on next page

Activate your InfoObject **CUST##** by choosing the *Activate* icon.

- a) From the context menu of your InfoObject Catalog choose *Create InfoObject* and in the *Create InfoObject* dialog enter the values:

<i>Char.</i>	CUST##
<i>Description</i>	GR## Direct Customer
<i>Template</i>	OCUSTOMER

- b) On the *Attributes* tab **deselect** all Navigational Attributes.
- c) On the *Master data/texts tab* change the *Master Data Access* to **Remote** from the selection list.
- d) Save your InfoObject CUST## by choosing the *Save*  icon.
- e) Activate your InfoObject CUST## by choosing the *Activate*  icon.
3. In order to make your Characteristic available for direct access, your InfoObject **CUST##** must be inserted into the InfoProvider tree as an InfoProvider.
 - a) Use the menu *Data Warehousing Workbench: Modeling* → *InfoProviders* → *InfoArea: BW Training* → *BW Customer Training* → *BW350 BI Data Acquisition* → *BW350 Group ## (T_BW350_GR##)* and from the context menu of your InfoArea choose *Insert Characteristic as InfoProvider*.
 - b) In the dialog box, type in your InfoObject **CUST##**. Your InfoObject will then be inserted into the InfoArea as an InfoProvider. Expand the InfoProvider to see that the text InfoProvider **TEXTS CUST##** is also present.
4. Create a Direct Access **DataSource DB_DCUST##** for Source System **ZMSS_00** (DB Connect Source System) under your Application Component **ZT_BW350_GR##**.

<i>DataSource</i>	DB_DCUST##
<i>Data Type DataSource</i>	Master Data Text

On the *General Info.* tab enter **GR## Direct Access** for the short, medium and long descriptions.

On the *Extraction* tab, select **Allowed** from the list in the field *Direct Access*.

Continued on next page

This DataSource will access the requested master data texts from a DB Connect Source (Data Base View) which is connected to the BI system. In the field *Table/View* field enter the data base view name **RUDICUSTOMERS**.

- a) Use the menu *Data Warehousing Workbench: Modeling* → *Source Systems* → *DB Connect* → *ZMSS_00* and from the context menu choose *Display DataSource Tree*.
- b) In the DataSource tree, find your application component **ZT_BW350_GR##** and from its context menu choose *Create DataSource*.
- c) In the dialog *Create DataSource* enter the following values:

<i>DataSource</i>	DB_DCUST##
<i>Data Type DataSource</i>	Master Data Text

- d) On the *General Info.* tab enter **GR## Direct Access** for the short, medium and long descriptions.
- e) Do not forget to select **Allowed** from the list in the field *Direct Access* on the *Extraction* tab.

This DataSource will access the requested master data texts from a DB Connect Source (Data Base View) which is connected to the BI system. In the field *Table/View* field enter the data base view name **RUDICUSTOMERS**.

In the field *Table/View* use the selection list and then choose the *Execute*  icon to see the list of database tables and views.

- f) In the list of database tables and view, select the database view **RUDICUSTOMERS** and choose the *Continue (Enter)*  icon to return to the DataSource.

Check the *Proposal* and *Fields* tabs to see the list of the fields from the database view.

Then activate your DataSource by choosing the *Activate*  icon.

- g) Go to the *Preview* tab and choose the *Read Preview Data* to see the text data.

5. Create a Transformation between your DataSource **DB_DCUST##** and your InfoProvider **TEXTS CUST##** and map the following fields.

Continued on next page

<i>CUSTID</i>	CUST##
<i>CUSTNAME</i>	0TXTMD

- a) From the context menu of your InfoProvider **TEXTS CUST##** choose *Create Transformation*.

In the dialog *Create Transformation* enter the following values for *Source of the Transformation*:

<i>Object Type</i>	DataSource
<i>DataSource</i>	DB_DCUST##
<i>Source System</i>	ZMSS_00

Choose the *Continue (Enter)*  icon.

- b) In the *Create Transformation* work area draw a line between the following fields:

<i>CUSTID</i>	CUST##
<i>CUSTNAME</i>	0TXTMD

Then choose the *Activate*  icon.

6. Create a direct access Data Transfer Process for your InfoProvider **TEXTS CUST##**.

- a) Expand your InfoProvider **TEXTS CUST##** to find the Data Transfer Process **TEXTS CUST##** and from its context menu choose *Create Data Transfer Process*.
- b) In the dialog *Create DataTransfer Process* accept all the defaults and choose the *Continue (Enter)*  icon.
- c) Ensure that the *DTP Type* is **DTP for Direct Access** and that on the *Extraction* tab the entry for *Table/View* is **RUDICUSTOMERS**.

Then choose the *Activate*  icon to activate your Data Transfer Process.

Continued on next page

7. Check the results of the direct access by using transaction LISTCUBE for direct access of master data texts.
 - a) Use transaction LISTCUBE and in the field *Data Target* enter **CUST##\$T** and then choose the *Execute* icon.
 - b) Choose the *Field Selection for Output* button and select the following fields:

X	<i>Medium description</i>
X	<i>GR## Direct Customer</i>

Then select the *Execute* icon and then will return you to the previous screen.

- c) Then select the *Execute* icon again.

Result

Congratulations now you see the contents of the database view by direct access.



Lesson Summary

You should now be able to:

- Describe how to access data in BI source systems directly

Lesson: Real Time Data Acquisition

Lesson Overview

Real time data warehousing is a framework for deriving information from data as the data becomes available. It features a lower time scale than for scheduled/batch data acquisition (hourly as compared to daily) and provides near immediate availability of data for reporting. In general, real time data warehousing supports tactical decision-making.



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the real time data acquisition process
- Describe real time data acquisition with the Service API
- Describe real time data acquisition with web services

Business Example

Under certain circumstances, your company would like to set up reporting using sales and purchasing data from your Source Systems in BI on a near immediate basis. You have created the relevant DataSources in BI to enable this real time data acquisition for reporting.

Real Time Data Acquisition

Real-time data acquisition supports tactical decision making. In terms of data acquisition, it supports operational reporting by allowing you to send data to the delta queue or PSA table in real time. You use a daemon to transfer DataStore objects that have been released for reporting to the DSO layer at frequent regular intervals. The data is stored persistently in BI.

You use real-time data acquisition if you want to transfer data to BI at frequent intervals (every hour or minute) and access this data in reporting frequently or regularly (several times a day, at least). The DataSource has to support real-time data acquisition. The option to support real-time data acquisition is a property of a DataSource.



Note: DataSources that are released for real-time data acquisition cannot be used for standard data transfer (scheduled staging).

Real-time data acquisition can be used in two primary scenarios:

- via the Service API (SAPI). It incorporates usage of InfoPackage for Real-time Data Acquisition (source to PSA). It then leverages Data Transfer Process for Real-time Data Acquisition (PSA to DataStore Object).
- via a Web Service Incorporates usage of Web Services to populate the PSA which then leverages the Real-time DTP to transfer data to the DataStore Object

The following graphic illustrates the process flow for the real-time data acquisition using SAP applications:

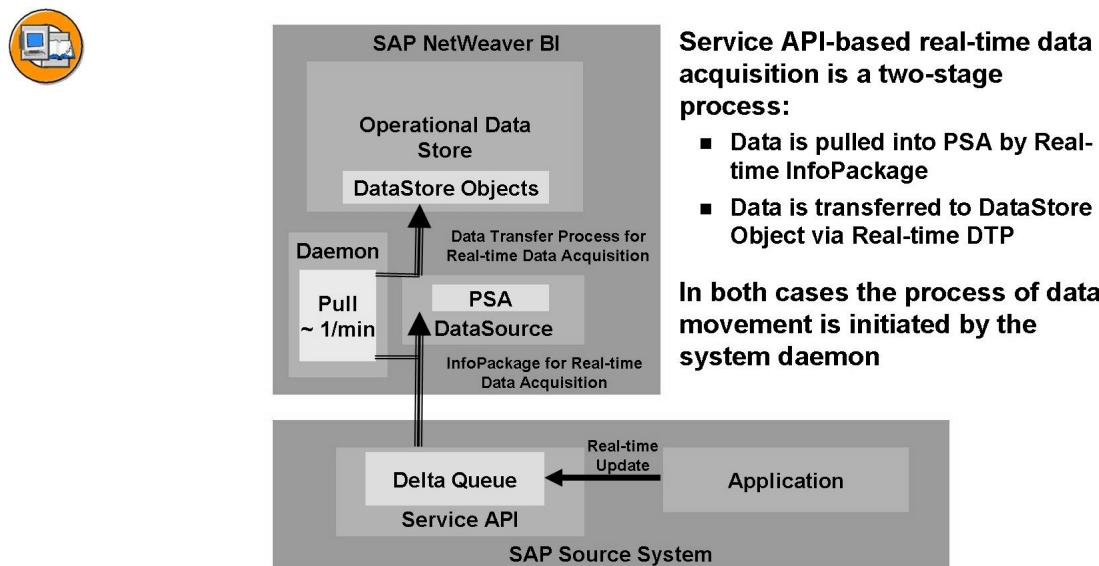


Figure 46: Real Time Data Acquisition from SAP Systems

The following graphic depicts real-time data acquisition with web services:

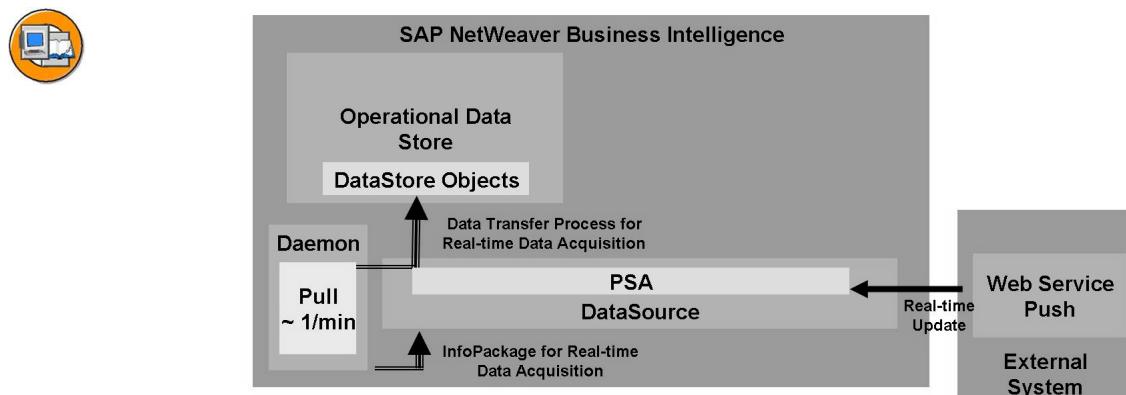


Figure 47: Real Time Data Acquisition using Web Services

Data is loaded into BI at frequent, regular intervals and is then posted to the DataStore objects that are available for operational reporting. In BI, special InfoPackages are created for real-time data acquisition.

These are scheduled using an assigned daemon and are executed at regular intervals. With certain data transfer processes for real-time data acquisition, the daemon takes on the further posting of data to DataStore objects from the PSA. As soon as data is successfully posted to the DataStore object, it is available for reporting. Refresh the query display in order to display the up-to-date data. The query displays the time that the data was last refreshed by a daemon run. Even if no new data has been posted, the query displays the time of the last daemon run. You can transfer data from the source to the entry layer of BI (the PSA) in two ways:

1. You use the Web service to push the data from the source into the PSA directly. In this case, the transfer of data is controlled externally, without placing demands on BI. An InfoPackage (for full upload) is only required in order to determine specific parameters for real-time data acquisition.
2. When the delta queue in the source system requests data, the data is loaded into the PSA using an InfoPackage created specifically for this purpose. You have to simulate the initialization of the delta process for the DataSource beforehand. The following two scenarios are possible:
 - a) The application in the source system writes the data to the delta queue (for example, V3 update for LIS DataSources). In this case, the daemon fetches the data without calling the extractor.
 - b) The data is not automatically written to the delta queue. With generic extractors or extractors that transfer data synchronously from BI to the Service API on request, the daemon calls the extractor and the extractor writes the data to the delta queue. The data is transferred to BI directly from the delta queue.

The requests for InfoPackages and data transfer processes are synchronized and remain open throughout several load processes. They are only closed when the size limits and time limits set in the InfoPackage are exceeded. The data transfer process copies these settings from the InfoPackage. When requests are closed, new requests are opened automatically and the data transfer for real-time data acquisition continues with these new requests. The data (and the open requests) are available in reporting as soon as they have been successfully posted to the DataStore object.



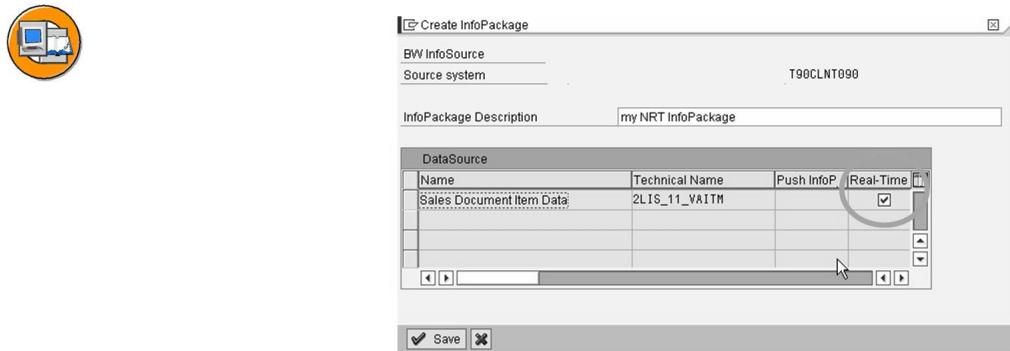
Define the data model for real-time data acquisition.

1. Create the DataSource and activate it in BI.
2. Create a DataStore object.
3. Create a transformation.
4. Define a data transfer process for real-time data acquisition.
5. Create an InfoPackage for the DataSource for simulating the delta initialization. You only need this InfoPackage if you are loading data into BI using the Service API.
6. Create an InfoPackage for the DataSource for real-time data acquisition. If you are transferring data using the Web service, you only need this type of InfoPackage. The InfoPackage applies the threshold values that are required for real-time data acquisition to the Web service.
7. Switch from the InfoPackage to the monitor for real-time data acquisition. In the monitor, you define the daemon and assign a DataSource (and InfoPackage) and data transfer process to it.

Daemon for Real-Time Data Acquisition

A **daemon** is a background process that processes the InfoPackages and data transfer processes assigned to it at regular intervals. It is used to control the transfer process for real-time data acquisition. The daemon receives information from the InfoPackage and data transfer process about which data is to be transferred or updated further, which PSA tables or DataStore objects are to be filled, and when a request should be closed and a new one opened.

When you extract data, the daemon works on the basis of the list of DataSources assigned to it in the InfoPackage, extracts the data from the source systems, and transfers it to the PSA table and DataStore objects. The InfoPackage definition for SAP applications is shown below:



Real-time flag must be selected for the InfoPackage which the daemon will monitor.

This option is only available for one InfoPackage attached to the DataSource

Note: If you want to use a DataSource with real-time capabilities, you have to install the PI_BASIS 2005.1 in the source system

Figure 48: InfoPackage for Real-time Data Acquisition Creation

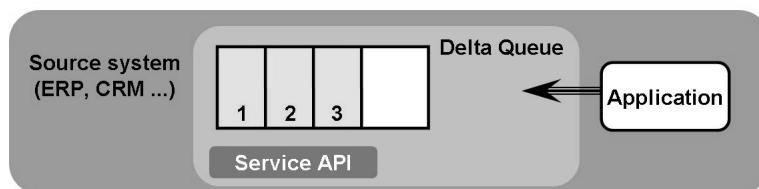
It informs the Service API in the source system when the data for the source system has been successfully updated. When the PSA request has been closed successfully and a new delta request has been opened, the updated data is deleted from the delta queue of the source system.

When you further update data to the DataStore objects from the PSA, the daemon works on the basis of the list of sources (DataSources) and targets (DataStore objects) assigned to it in the data transfer process. It transfers data to the DataStore object. The data is written straight to the A table and the change log. If the extraction or update is terminated, the daemon can restart the process from the point at which it was terminated. In the case of data transfer processes, you can determine the maximum number of errors that are allowed before the process is finally terminated. The following graphics depict the process of extracting data every three minutes:

The following graphics illustrate the activities on the SAP application side:



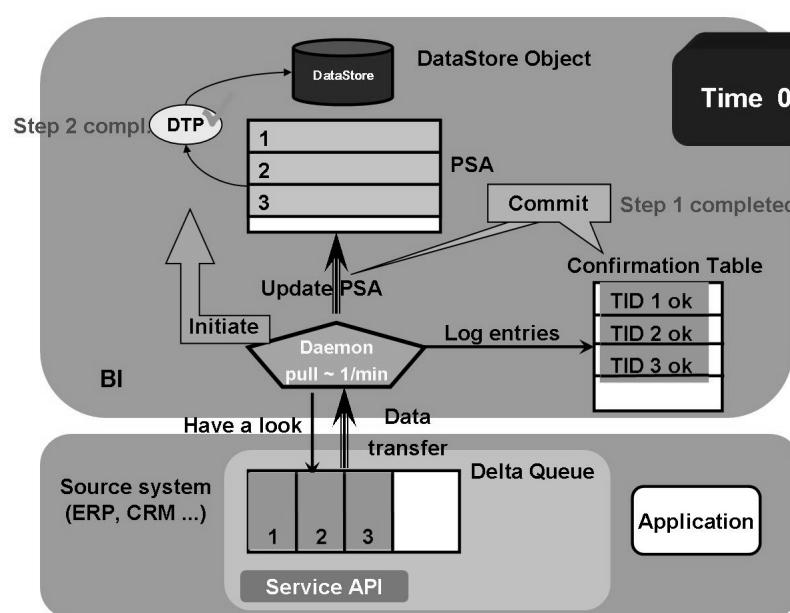
Time 03:10:09

**Figure 49: SAP Application Source System Activities**

The Daemon then triggers Realtime Data Acquisition processing from the application as illustrated below:



Time 03:11:00

**Figure 50: Daemon Triggers RDA Processing**

The following two graphics outline the steps the Daemon initiates:

**Step 1**

- Call source system for new records
- Status update of transferred records in confirmation table
- Update PSA
- Commit
 - ◆ Update confirmation table and update PSA in one step
 - ◆ Guarantees synchronization

Figure 51: RDA Runtime Step 1

After the confirmation table and the PSA are updated:

**Step 2**

- Check for records in confirmation table
 - ◆ If records are available in confirmation table the corresponding records exist as well in PSA
- Daemon flags records in source system
- Reply confirmation sent to SAP NetWeaver BI
- Entries are flagged as processed in confirmation table
- Initiate DTP after the records are confirmed in confirmation table
- Commit
 - ◆ If not successful in BI, records will stay in confirmation table
 - ◆ Guarantees restart of step 2 even when the update in the source system was successful

Figure 52: RDA Runtime Step 2

The following illustrates the Daemon triggering RDA processing three minutes after the first initiation:

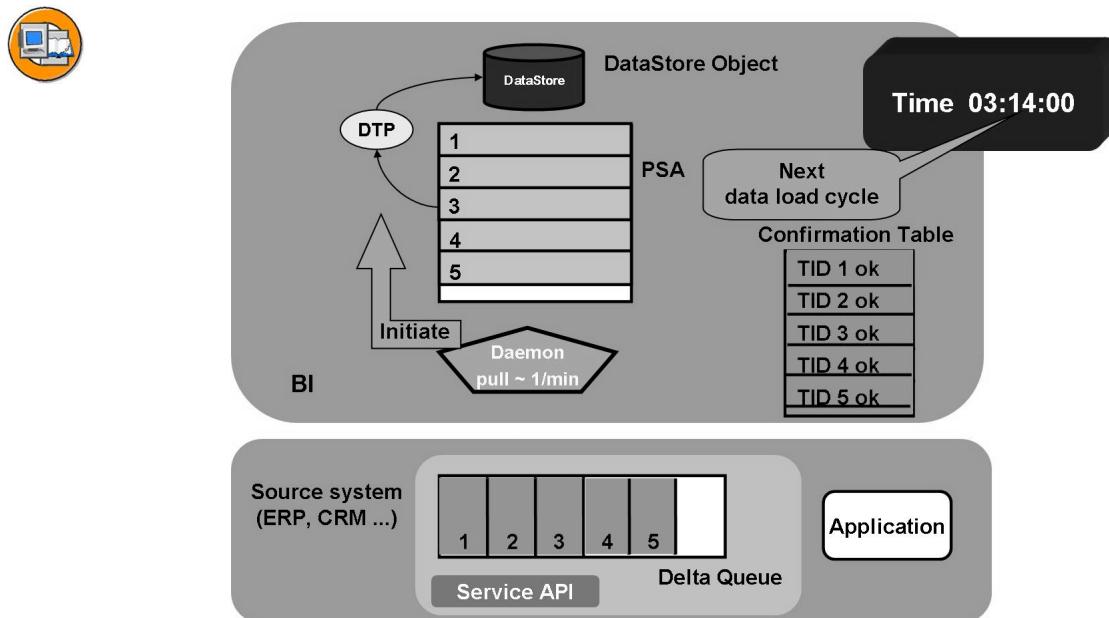


Figure 53: Daemon Triggers RDA Processing Every 3 Minutes

Once the data load into the PSA is complete, then the open request is closed.

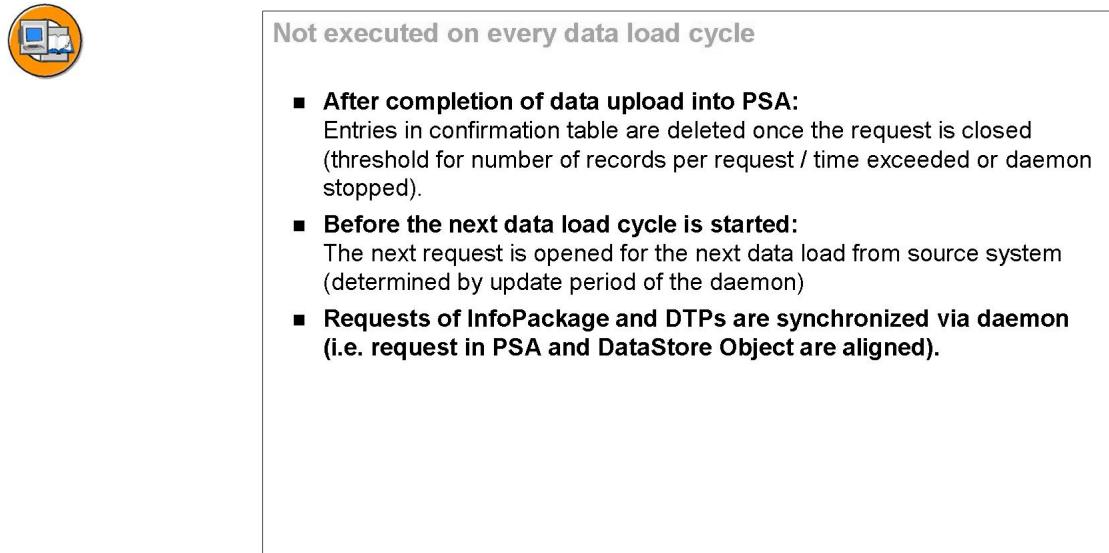


Figure 54: Closure of Open Request

The Daemon then closes the request.

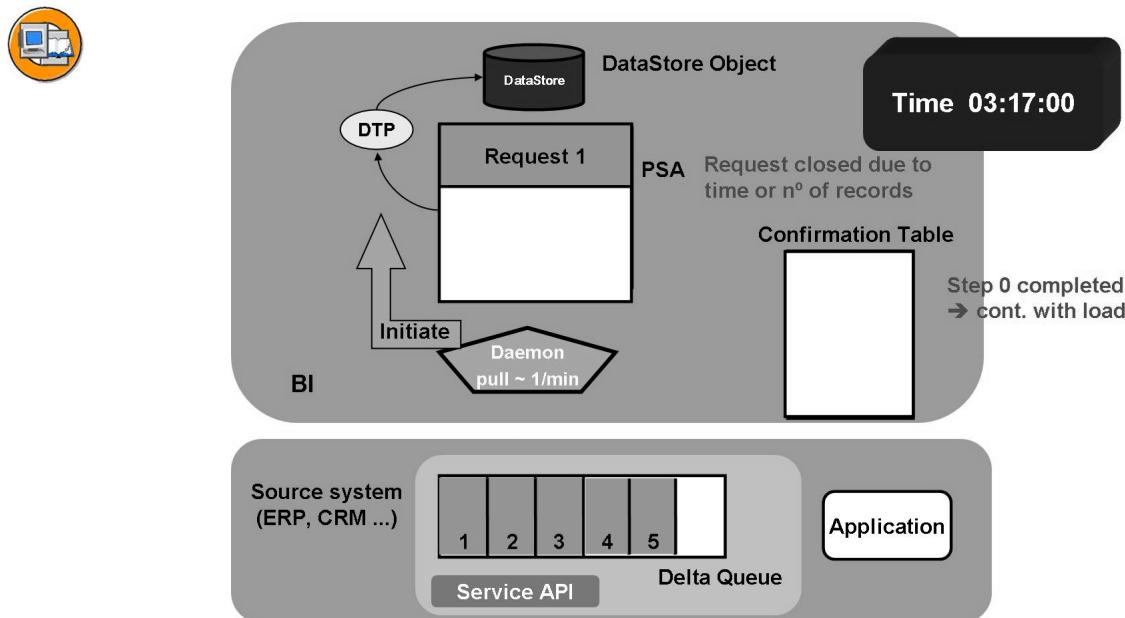


Figure 55: Daemon Closes Request

The daemon runs in a permanent background job and only switches to "sleep" mode after performing a data transfer if there is currently no other data in the delta queue and no other data being posted to the PSA by a Web service. If the Service API is being used to transfer the data, the RFC connection to the source system remains open. This implies that a permanent RFC connection between every source system and BI is required for real-time data transfer using the Service API.

→ **Note:** So that the system does not use too much main memory and the RFC connection does not have to be open for too long, the daemon stops independently on a regular basis and schedules again, so that the main memory can be released and a new RFC connection is opened. This happens without having to close the request for real-time data acquisition

Monitor for Real Time Data Acquisition

In the monitor for real-time data acquisition (transaction RSRDA), the daemons that are currently available in the system are displayed in a tree structure. InfoPackages and data transfer processes (DTP) are displayed under the relevant DataSources.



Monitor: Real-Time Data Acquisition

Daemon/Info Pack./InfoProv./Req. ID	DataStore Object	Period	Last Upload	Last Upload	Records in Request
Daemon Nr01					
Daemon Nr03					
Daemon Nr05					
PM_RDA					
QAO DTP_3Y9VXF4WD\$6BCYGYE6F	FWL_NMM04	10 min			
QAO ZPAK_3Y9VIT4FAH3N0484BB	ZDICB0002440	10 min			
QAO REQU_3YLB4VF9R41RTY	/BIC/B000244U				

DataSource **Real-time InfoPackage** **Real-time DTP**

The RDA Daemon monitor provides an overview on the status of each daemon and attached

- InfoPackage(s) for RDA
- Data Transfer Process(es) for RDA

Important to note that both loading to the PSA and the DataStore Object are monitored within the Daemon Monitor

Transaction RSRDA

Figure 56: RDA Daemon Monitor

The monitor displays the status of the daemons and the assigned InfoPackages and data transfer processes, along with their open requests. It also includes the monitor functions for processing the daemons and the InfoPackages and data transfer processes. In the *Administration* functional area in the Data Warehousing Workbench, choose *Monitors* to access the real time monitor.

Under a daemon, the system displays the DataSources that you can use to load data with real-time data acquisition that already have an InfoPackage for real-time data acquisition. DataSources with InfoPackages that are not assigned to a daemon are displayed under *Non-Assigned Nodes*. Under a DataSource, the system displays the InfoPackage that contains the parameters for the data transfer, along with the data transfer process that uses this DataSource to fill a DataStore object. Under an InfoPackage or data transfer process, the system displays the open requests that post data to the PSA table or DataStore object. You can update the overview using *Extras → Refresh*.

Data is transferred real time into BI on request using the Service API

An InfoPackage for real-time data acquisition starts the transfer of data from a DataSource. Data is pulled into the PSA by a real time InfoPackage and then is transferred to a DataStore object with a real time DTP.

If you use the Service API in BI to transfer data, use the InfoPackage to determine the parameters for the data request. For real-time data acquisition, specify (in the InfoPackage) how the daemon is to process the InfoProvider.

If you are using a Web service to transfer data into BI, you also need an InfoPackage to determine the size limit and time limit for a request (when this threshold value is exceeded, the system closes the request and continues to transfer data transfer with a new request). You also use the InfoPackage to determine the size of the data package (this may affect performance when you post data to the PSA).

The involved daemon then performs three steps for the InfoPackage assigned to it

1. In the first step, the daemon calls the Service API in the source system. The Service API transfers the data records into BI (via the delta queue, where necessary) and passes them to the daemon. The data is updated in the PSA table.
2. If the data has been successfully updated to the PSA table, in the second step, the daemon confirms the transfer and the Service API changes the status of the records in the delta queue. Once the request has been successfully closed and the next request is open, the records are deleted from the delta queue.
3. In the third step, the data transfer process for real-time data acquisition is started. This updates the data from the PSA table into the DataStore object. The changes to the A table of the DataStore objects are logged in the appropriate change log requests. The request that transfers the data to the PSA table (PSA request), the data transfer process request, and the change log request for each DataStore object have a 1:1 relation. The data is written directly to the A table and the change log. An additional step for activating the data in the DataStore object is not required. Since the data is activated immediately, data is not written to the activation queue of the DataStore Objects.



Creating an InfoPackage for Real-Time Data Acquisition

Prerequisites

The DataSource for which you want to create an InfoPackage supports real-time data acquisition. If you are using the Service API to transfer data into BI, you have already created an InfoPackage for this DataSource. This InfoPackage has performed a simulated initialization of the delta process.



Note: You can only create **one** InfoPackage for a DataSource for real-time data acquisition

Procedure

1. In the modeling view of the Data Warehousing Workbench, go to the DataSource tree of the required source system, select the DataSource, and in the context menu, choose *Create InfoPackage*.
2. In the following dialog box, enter a description for the InfoPackage. In the *DataSource* table, select the required DataSource and select *Real-Time Data Acquisition InfoPackage* for the DataSource. This field is only input ready if the DataSource supports real-time data acquisition and no InfoPackage for real-time data acquisition has been created for this DataSource.
3. Choose *Continue*. You get to the scheduler, where you can maintain InfoPackages.
4. Tab page *Extraction*: If you are using the Service API to transfer data, *Real-Time Extraction from SAP Source System* has to be defined as the adapter. If you are using the Web service to transfer data, *Web Service (Push)* has to be defined as the adapter.
5. Tab page *Processing*: Here you determine the number of days and hours after which the request should be closed (independent of the number of records loaded); the default setting is one day.

In addition, you determine how many data packages are allowed for each request (default setting: 50), and the maximum number of searches for errors the system is to perform before a request is evaluated as incorrect. The default setting for data package size in the data transfer before a request is closed is 10,000 records. Only the debug user can change this setting. If a request is closed based on the settings you have made, the system automatically starts a new request to continue the data transfer.

6. Tab page *Scheduling*: Choose *Assign* to access the monitor for real-time data acquisition.

Continued on next page

Result

In the monitor for real-time data acquisition, the system displays the InfoPackage in the *Non-Assigned Node* area, under the DataSource for which the InfoPackage has been created. In the context menu of the DataSource, choose *Assign Daemon* to assign the InfoPackage to a daemon. The daemon can use the InfoPackage for data extraction.



Creating a Data Transfer Process for Real-Time Data Acquisition

Use

You use the data transfer process for real-time data acquisition to transfer data to the DataStore object from the PSA. In the DataStore object, the data is available for use in reporting.

Prerequisites

You have created the dataflow between the DataSource and the DataStore object using transformations. The selections for the data transfer process do not overlap with selections in other data transfer processes.

Procedure

1. In the Data Warehousing Workbench, an object tree is displayed and you have highlighted the DataStore object. and in the context menu, choose *Create Data Transfer Process*.
2. In the dialog box for creating a data transfer process, . Choose *DTP for Real-Time Data Acquisition* as the DTP Type.
3. As the source object, select the DataSource from which you want to transfer data to the DataStore object. Then choose *Continue*. The data transfer process maintenance screen appears. The header data for the data transfer process shows the description, ID, version, and status of the data transfer process, along with the delta status
4. On the *Extraction* tab page, determine the parameters: *Delta* is chosen as the extraction mode for real-time data acquisition. If necessary, determine filter criteria for the delta transfer.



Note: This means that you can use multiple data transfer processes with disjunctive selection conditions to transfer small sets of data from a source efficiently into one or more targets, instead of transferring large volumes of data. You can specify individual selections, multiple selections, intervals, selections based on variables, or routines. Choose *Change Selection* to change the list of InfoObjects that can be selected.

Continued on next page

5. On the *Update* tab page, determine the parameters:
 - Apply settings for error handling
 - Determine how the system updates the valid records when errors occur
 - The number of errors allowed before the load process terminates
 - Whether the system gives error status to a load process if records are aggregated, filtered out, or added in the transformation
- **Note:** Note that error handling settings only have an impact while repairing a DTP request (repair) and during the conversion of the DTP to standard DTP (for example, to correct an error during extraction). For real-time data acquisition, specify in the InfoPackage the maximum number of failed attempts that is permitted when the daemon is accessing data before the data transfer terminates with an error.
6. On the *Execute* tab page, which depicts the process flow of the program for the data transfer process in a tree structure, determine the parameters:
 - Specify the status that you want the system to adopt for the request if warnings are displayed in the log.
 - Specify how you want the system to determine the overall status of the request.
7. Check, save and activate the data transfer process.

Result

Choose *Real-Time Monitor* to access the monitor for real-time data acquisition. In the monitor, you can assign the data transfer process to a DataSource and a daemon. You do this in the context menu of the DataSource. The daemon uses the data transfer process to process data.

Data is transferred real time into BI on request using Web Services

Web Services-based real time data acquisition is also a two stage process:

1. Data is pushed into the PSA using a Web Service
2. Data is transferred to a DataStore object using a real time DTP

The following graphic illustrates the real time data acquisition process using Web Services.

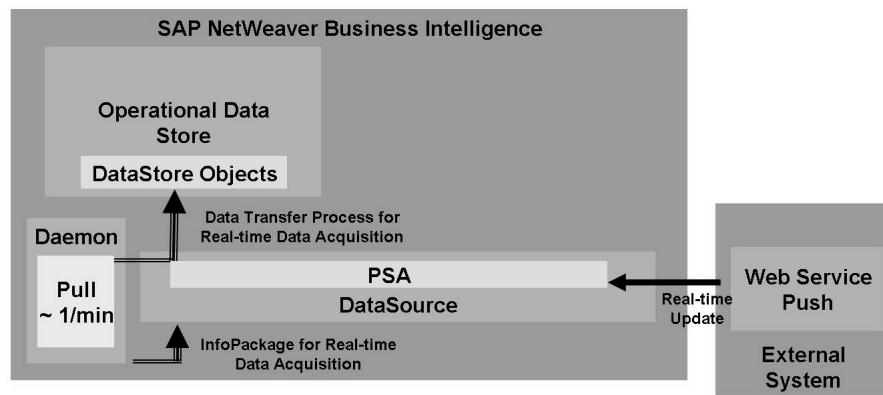
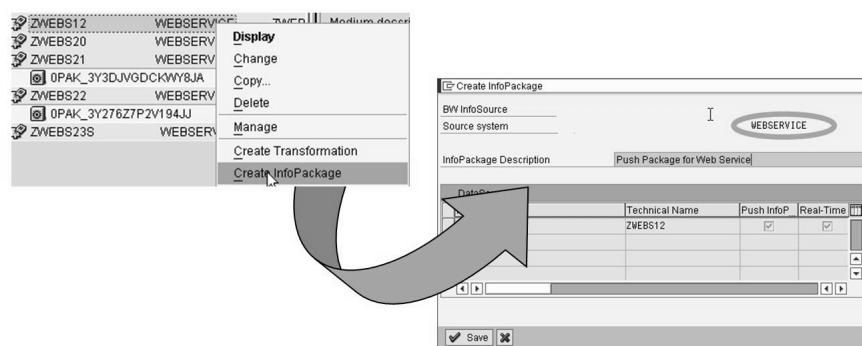


Figure 57: Real Time Data Acquisition via Web Services

A daemon monitors the PSA and initiates the DTP on a regular basis according to settings in a real time InfoPackage. The InfoPackage is needed for the Web Service to define parameters such as time / record thresholds for request closure and to allow the assignment of a daemon using the Real time Monitor as seen below:



An InfoPackage is needed for the Web Service to define parameters such as time / record thresholds for request closure and to allow the assignment of a daemon via the Real-time Monitor.

Figure 58: InfoPackage Definition for RDA with Web Services



Lesson Summary

You should now be able to:

- Describe the real time data acquisition process
- Describe real time data acquisition with the Service API
- Describe real time data acquisition with web services



Unit Summary

You should now be able to:

- Describe the data transfer process
- Contrast the new data flow with the previous data flow
- Describe the conversion path to the new data transfer process
- Describe the transformation process
- List the functions available for transformation of data during extraction
- Describe how to access data in BI source systems directly
- Describe the real time data acquisition process
- Describe real time data acquisition with the Service API
- Describe real time data acquisition with web services

Unit 3

Data Acquisition with the Service API

Unit Overview

In this unit, you will learn about the technology behind SAP source system linking and get to know the various data transfer methods and their control parameters. This unit then introduces the main role of the BI Service API for data extraction. In addition to the transfer of Business Content DataSources, you will learn how to use the tools for defining generic DataSources. At the end of this unit, the logistic data extraction and the required settings in LO BI Customizing Cockpit will be explained in more detail as an example of BI Content extractors. You will also get an overview of the options for enhancement of DataSources.



Unit Objectives

After completing this unit, you will be able to:

- Describe the basic principles for connecting SAP source systems to a BI system
- Describe the role of the BI Service API in data acquisition, extraction and staging
- Describe the transfer process for Business Content DataSources
- Describe how the system decides which DataSource type has to be generated
- Describe the necessary steps to extract data from the Logistics application
- Explain the functions in the Logistics Extraction Structure Customizing Cockpit
- Explain the role played by the restructuring (setup) tables and the delta queue in Logistics data extraction
- Describe the update methods to fill the delta queue
- Create generic DataSources
- Use different data acquisition methods to create generic DataSources
- Describe the emulation process of existing generic and non-generic DataSources
- Enhance Business Content DataSources according to the demands of your company

- Implement the data enhancements using the function enhancements of the service API
- Determine the function enhancements in the individual applications

Unit Contents

Lesson: Connecting SAP Source Systems to a BI System	145
Lesson: Basics of the BI Service API	150
Exercise 4: Basics of the BI Service API (Optional)	157
Lesson: Transfer of Business Content DataSources.....	162
Lesson: Logistics Data Extraction	166
Exercise 5: Overview of the LO Extraction (LO Cockpit).....	185
Lesson: Generic Data Acquisition	192
Exercise 6: Generic DataSources for Attributes/Texts (Optional)....	199
Lesson: Enhancement of Business Content DataSources.....	219
Exercise 7: Enhancing a DataSource for Transaction Data	227

Lesson: Connecting SAP Source Systems to a BI System

Lesson Overview

In this lesson you will learn the basic principles for connecting SAP source systems to a BI system.



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the basic principles for connecting SAP source systems to a BI system

Business Example

Your company would like to set up reporting on sales and purchasing data from your SAP source system in BI . To enable the reporting, you first have to identify how to link the SAP source system to the BI system.

Connecting SAP Source Systems to a BI System

An SAP source system is created in the Data Warehousing Workbench.

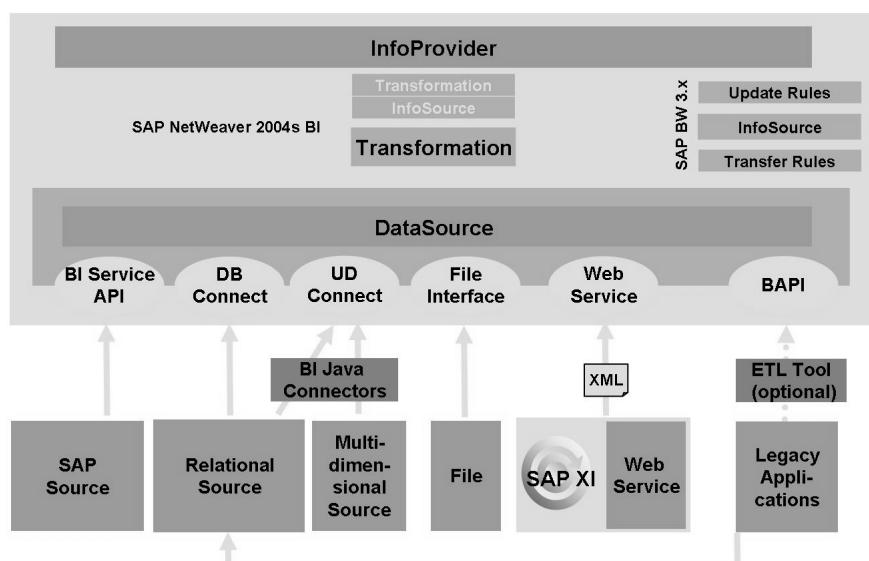


Figure 59: Extracting Data from SAP Source Systems

The data is extracted from the SAP source system and updated using a DataSource. The client in an SAP component is always connected as a source system in BI.

The following selection options are available when you create a source system for BI (see also the below graphic):

1. SAP systems

When you select automatic creation, the required RFC connections are generated at the same time (recommended.) If you choose to create the source system manually, you have to create the RFC connections between the BI system and the SAP source system first.

2. BI systems

3. Flat files for which metadata is maintained manually and transferred to BI using a file interface

4. Database management systems into which data is loaded from a database supported by SAP using DB Connect, without using an external extraction program

5. Relational or multidimensional sources that are connected to BI using UD Connect

6. Web Services that transfer data to BI by means of a push

7. Non-SAP systems for which data and metadata is transferred using staging BAPIs



Note: This functionality only works in conjunction with DataSource 3.x and not the new DataSources.

Source Systems	Tech. Name	M...	Execute Function	Display Tree	Ob...	Object Informati...	Object I...
BI			Change				
Myself-system	FB4CLNT800		Display DataSour...				
ID3 Client 800	T90CLNT090		Display DataSour...				
SAP	SAP		Change				
External System	PARTNERS		Change				
File	FILE		Change				
DB Connect	DB		Change				
UD Connect	UDC		Change				
Web Service	WEB		Change				

Automatic Create SAP System
 Manually Create SAP System
 SAP Business Information Warehouse
 File System (Manual Metadata, Data Using File Interface)
 Database System (Data and Metadata Using SAP DB Connect)
 SAP UDC System (Data and Metadata with UDC and Portal Server)
 WebService System (Metadata Manually, Data with WebService Push)
 External System (Data and Metadata Transfer Using Staging BAPIs)

Figure 60: Setting up Source Systems in BI

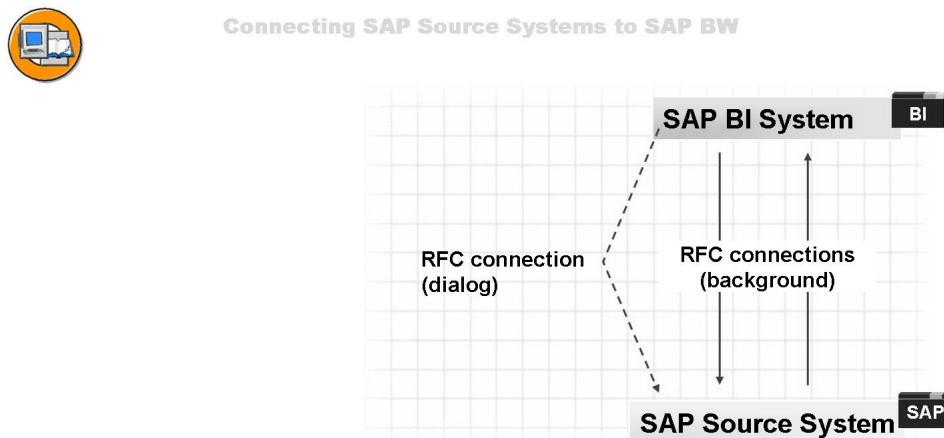
You define source systems in the Data Warehousing Workbench in the source system tree. To define a source system, choose *Create* in the context menu of the folder in the source system type.

BI is a data warehouse solution and as such it is closely linked in a network to other systems. Usually, there are several source systems connected to a BI system, but the BI system itself can also act as a source system to another BI system. In this case, the term data marts is used (see unit 9 *Data Acquisition Using the Data Mart Interface*). Since modifications to a system that is part of a source system/BI connection have an impact on all the connected systems, they cannot be treated in isolation.

A connection between a source system and an BI system consists of a series of individual connections and settings that are made in both systems:

- *RFC Connections*
- *ALE Settings*
- *Partner Profiles*
- *Ports*
- *IDoc Types*
- *IDoc Segments*
- *BI Settings*

The section entitled *Configurations in BI* in the BI documentation contains more important information on this topic.



RFC (Remote Function Call) connections are created between the systems

A user is required in each system to enable the systems to communicate with one another

Based on the logical system name of the client in a SAP source system

Figure 61: Connecting SAP Source Systems to an BI System

You use transaction *SM59* to maintain RFC connections. RFC connections are based on ALE technology (Application Link Enabling). ALE is a technology for the construction and operation of distributed applications. It provides for the efficient and controlled exchange of messages and keeps data consistent in loosely connected SAP application systems. The applications are integrated using *synchronous* and *asynchronous* communication, rather than a central database.

RFC connections are created based on the logical system name. This allows a *client* to be uniquely identified in an *SAP system landscape*.

A second RFC connection (with *_DIALOG* as the suffix to the technical name) is created to enable you to jump from the BI system to the SAP source system during online processing (for example, out of the monitor assistant, or to make Customizing settings in the source system). No user is specified in this connection. BI users must therefore log on their source system user names and passwords before jumping to the SAP source system. You will find detailed up-to-date information on this topic in the SAP Service Marketplace.

Note the following with regard to background users:



Hint:

1. The default for the user should be defined in the source system in BI Customizing (transaction *SPRO*) to ensure that it is defined correctly.

Menu path: *BI Customizing Implementation Guide* → *Business Information Warehouse* → *Connections to Other Systems* → *Connections Between SAP Systems and BI*

2. The following basic points apply:

Both background users should be of the *system* user type. There are special authorization profiles for both background users. This is the only way to prevent these users from being used incorrectly. In the RFC connection, the current password for the background user must be entered in the SAP source system that is going to be connected to the BI system.



Lesson Summary

You should now be able to:

- Describe the basic principles for connecting SAP source systems to a BI system

Lesson: Basics of the BI Service API

Lesson Overview

In this lesson you will learn about the basics of the BI Service API with regard to the extraction.



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the role of the BI Service API in data acquisition, extraction and staging

Business Example

Your company would like to set up reporting in BI using sales and purchasing data from your SAP source system. It is therefore important that you understand the functions of the Service API, such as data and metadata transfer from SAP source systems.

BI Service API

The BI service API (Application Programming Interface) is based exclusively on SAP technology and is used at various points within the BI architecture:

- To transfer data and metadata from SAP source systems
- With XML/SOAP-based data transfer (see unit 8)
- With data transfer using the Data Mart Interface (see unit 10)

The following section provides an overview of the main functions and tasks of the BI Service API:



- Configuration of source system connections
- Delta Queue (clipboard for delta records)
- Data extraction monitored by exchanging messages (status information) between systems (Info-IDocs in monitoring)
- Less downtime at initialization (system data is mirrored)

Figure 62: SAP Service API: Scope (1)

Delta queue

The delta queue is a central temporary repository for delta records (= new and changed records) in the SAP source system. You will find detailed information on the structure and function of the delta queue in unit 3 *Delta Management: Overview*.

Reduced downtime

You can also reduce downtime by executing delta initialization runs in *mirror systems*. There is a How to... guide for this purpose in the SAP Service Marketplace entitled *How to ... Minimize Downtime for Delta Initialization..*



- Customizing DataSources
- Replication of DataSources (meta data)
- Control parameters for data transfer
- RemoteCube technology (direct access)
- Test tool for the extraction of data (extractor checker)

Figure 63: SAP Service API: Scope (2)

Customizing DataSources

You customize the DataSources in transaction *SBIW* (*Customizing Extractors*):

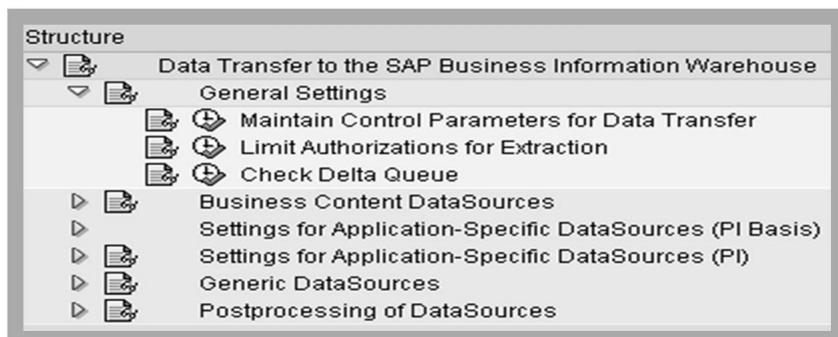


Figure 64: Customizing Extractors (SBIW)

- **Business Content: DataSources**

Before you can start to use SAP Business Content DataSources you have to transfer them from the Business Content. The transfer is made in two steps:

- Transfer of the application component hierarchy
- (Selective) transfer of DataSources

After the DataSources have been transferred, you can adapt them (by extending the extract structure, for example) and then replicate them in BI. The transfer and enhancement of Business Content DataSources will be dealt with in detail in the following lessons in this unit.

- **Generic DataSources**

You can use the following (user-defined) methods to perform the extraction, independently of a particular application:

- *DB Tables*
- *DB Views*
- *Function Modules*
- *InfoSet (SAP Query)*

No knowledge of ABAP programming is required for this. Generic DataSources enable data to be extracted to BI that neither the Business Content DataSources nor the application-specific DataSources would be able to provide. Detailed information is provided in the lesson *Generic Data Extraction* within this unit.

- **Technical properties of a DataSource**

- Extraction method/extractor
- Transfer method (*PSA* and *IDoc*)
- Delta process

If a DataSource is capable of producing delta figures, it describes a particular delta process in its metadata. The delta process indicates how data is transferred. It specifies how data is determined for the data target in BI. This enables you to work out, for example, the data targets for which a DataSource is suitable, how to update, and how the serialization has been done. You can find detailed information on the delta process in the unit *Delta Management: Overview*.

- Early Delta Initialization

If a DataSource has this property, you do not have to stop updating data in the SAP source system during delta initialization. Delta records can be posted at the same time as the initialization of a delta process for a DataSource is taking place. You can find detailed information on early delta initialization in the unit *Delta Management: Overview*.

- ...

(Global) control parameters for data transfer

You can maintain *control parameters for data transfer* in the transaction *SBIW* in the area *General Settings*. The parameters that you maintain here are valid as global standard values, and thus apply to all DataSources. However, for performance reasons there are Business Content DataSources with extractors that make their own settings. The individual parameters are described below:

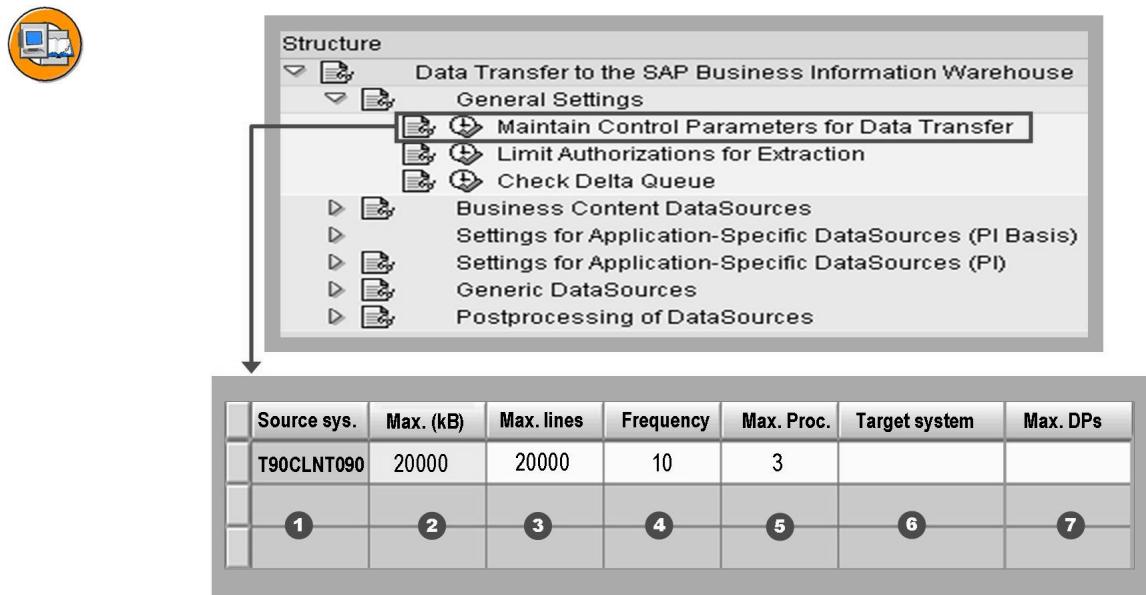


Figure 65: (Global) Control Parameters for Data Transfer

1. Source system

Enter the technical name of the logical system for your source client and assign the following parameters to it:

2. Max. (kB): Maximum size of data package

When data is transferred to BI, the individual data records are sent in packages of varying size. You use these parameters to set a maximum size for a data package. If you do not maintain an entry here, the data is transferred with a default setting of *10,000 kilobytes* per data package. The amount of memory required does not depend on the setting controlling the size of the data packages alone. It also depends on the length of the transfer structure, the memory requirement of the extractor involved, and with large data packages, it also depends on the number of data records contained in the package.

3. Max. rows: Maximum number of rows in a data package

In the case of large data packages, the memory requirement depends mainly on the number of data records transferred with this package. You use these parameters to set the maximum number of data records that a data package can contain. The maximum main memory requirement per data package is approximately $2 \times \text{Max. Rows} \times 1000 \text{ bytes}$.

4. Frequency

The frequency you set determines the number of data IDocs or data packages that have to be transferred before an InfoIDoc is triggered. The frequency is set to *1* by default. This means that an InfoIDoc is triggered after each data package. In general, you should choose a frequency between *5* and *10*, but not greater than *20*. The larger the data package, the lower you should set the frequency. In the monitor in BI, each InfoIDoc tells you whether the data is being loaded correctly or not.

5. Max. proc.: Maximum number of parallel processes for transferring data

This field is only available with release 3.1I or higher. Enter a number greater than *0*. The maximum number of parallel processes permitted at any one time is set by default to *2*. The optimal setting for the parameters depends on how the application server that you use to transfer data is configured.

6. Target system for batch jobs

Enter the name of the application server on which you want to run the extraction job.

7. Maximum Number of Data Packages in a Delta Request

You use this parameter to restrict the number of data packages in a delta request or in a repeated delta request. Use Only use this parameter if you are expecting delta requests with a very large volume of data so that more than 1000 data packages can exist in a request in spite of the large volume of data. No restriction is performed for the initial value or value *0*. Only a value greater than *0* leads to the number of data packages being restricted.



Caution: However, this number is not always adhered to. The actual restriction can deviate from the specified limit by up to *100*, depending on the extent to which data is compressed in the qRFC queue.



Hint: You can make DataSource-specific (local) adjustments to the parameters *Max. (kB)* and *Frequency* (in the context of the values set here) in InfoPackage maintenance:

*Maintenance screen: InfoPackage → Menu bar: Scheduler→ DataS.
Default setting for data transfer*

VirtualProvider: Technology

- *VirtualProvider based on Data Transfer Process for Direct Access*

An VirtualProvider is a InfoProvider type that enables you to define queries with direct access to transaction data in an SAP source system.



Hint: Not all DataSources are capable of providing data for VirtualProviders. This applies in particular to DataSources for using delta processes.

- *VirtualProvider based on BAPI*

A general VirtualProvider is a InfoProvider type with data that is managed externally to BI. A BAPI from a non-SAP system is used to read the data for reporting.

- *VirtualProvider based on Function Module*

A VirtualProvider based on Function Module is an InfoProvider type with data that is not managed in BI. This InfoProvider type enables you to construct analyses with the data from a function module that you have developed yourself, independent of the staging and the source systems.

Test tool for data extraction

As of BI Service API 3.0B you can test the extraction for DataSources in the SAP source system, independently of the BI system, with the transaction *RSA3 (extractor checker)*. This enables you to identify any problems before the actual extraction takes place. You can use this test tool for generic DataSources or DataSource enhancements too.

**Hint:**

1. *PI_BASIS* is a new software component (available from SAP Basis Release 6.20) that delivers the BI service API technology needed to connect to an BI system. A large part of the application extractors for Business Content are delivered with the software component *PI* (Business Content DataSources with extractors, extraction structures and so on).
2. You can display the current version of the BI Service API in the table *RSSAPIVERS* in the SAP source system too.
3. The *PI 2004.I* is the last SAP R/3 plug-in release to be delivered separately. New or enhanced interfaces for integrating *SAP R/3*, *SAP R/3 Enterprise* and *SAP ERP Central Component (SAP ECC)* will no longer be delivered as a separate add-on (SAP R/3 Plug-In). They are already included with the release SAP ECC 6.00 and higher.
4. You can find the latest information on the BI Service API and details of availability under <http://www.service.sap.com/bi> in the area *SAP BW Extractors/Plug-In*.

Exercise 4: Basics of the BI Service API (Optional)

Exercise Objectives

After completing this exercise, you will be able to:

- Display the basic configuration of the BI Service API in an SAP source system

Business Example

Your company would like to set up reporting in BI using sales and purchasing data from your SAP source system. It is therefore important that you understand the configuration of the Service API in preparation for data and metadata transfer from SAP source systems.

Task 1:

To display the basic configuration of the BI Service API in a SAP source system

1. In the BI system, execute transaction RSA1
2. In the *Data Warehousing Workbench*, select the functional area *Source Systems*.
3. Locate the source system, **T90CLNT090** and from the context menu go to the BI Implementation Guide in the source system.
4. In the *BI Implementation Guide*, open the node *General Settings* and find the IMG function *Maintain Control Parameters for Data Transfer*.
icon.
5. Fill out the grid with information concerning the source system **T90CLNT090**:

Option	Entry
Src. system	
Max. (kB)	
Max. lines	
Frequency	
Max. Proc.	
Target system for batch jobs	

Continued on next page

Task 2:

To display the structure and contents of the Delta Queue

1. Log directly into the source system **T90CLNT090**.
2. Access the Delta Queue.
3. Look at the contents of the entry for DataSource **2LIS_II_VAITM**.

Solution 4: Basics of the BI Service API (Optional)

Task 1:

To display the basic configuration of the BI Service API in a SAP source system

1. In the BI system, execute transaction RSA1
 - a) In the BI system, type the transaction RSA1 in the command field.
 - b) Select *Enter*.
2. In the *Data Warehousing Workbench*, select the functional area *Source Systems*.
 - a) Click on the *Source Systems* functional area.
3. Locate the source system, **T90CLNT090** and from the context menu go to the BI Implementation Guide in the source system.
 - a) Select the source system **T90CLNT090**
 - b) Right click on the source system and from the context menu choose the option *Customizing Extractors*
4. In the *BI Implementation Guide*, open the node *General Settings* and find the IMG function *Maintain Control Parameters for Data Transfer*.
icon.
 - a) Execute the IMG function *Maintain Control Parameters for Data Transfer* by clicking the  icon.
5. Fill out the grid with information concerning the source system **T90CLNT090**:

Option	Entry
Src. system	
Max. (kB)	
Max. lines	

Continued on next page

Frequency	
Max. Proc.	
Target system for batch jobs	

- a) For this particular source system, fill out the grid with the information provided in the configuration option.

Option	Entry
Src. system	T90CLNT090
Max. (kB)	20000
Max. lines	20000
Frequency	10
Max. Proc.	3
Target system for batch jobs	

Task 2:

To display the structure and contents of the Delta Queue

1. Log directly into the source system **T90CLNT090**.
 - a) Log directly into the source system with the logon information provided by your instructor.
2. Access the Delta Queue.
 - a) In the command field, insert the transaction RSA7 and press *Enter*.
3. Look at the contents of the entry for DataSource **2LIS_11_VAITM**.
 - a) Select the line containing the Data Source **2LIS_11_VAITM** and click the *Display* icon.



Lesson Summary

You should now be able to:

- Describe the role of the BI Service API in data acquisition, extraction and staging

Lesson: Transfer of Business Content DataSources

Lesson Overview

This lesson describes the steps that are required in order to use Business Content DataSources for staging data from SAP applications.



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the transfer process for Business Content DataSources
- Describe how the system decides which DataSource type has to be generated

Business Example

Your company would like to set up reporting using sales and purchasing data from your SAP source system in BI. You have selected the relevant Business Content DataSources in the SAP source system and want to transfer them.

Transfer of Business Content DataSources

The Business Content for SAP ECC and SAP source systems is a Plug-In component that creates the technical preconditions for the extraction. With BI systems (these systems can be connected as source systems to another BI system too), the Business Content is a fixed component in the system and does not have to be installed later by means of plug-ins.

The DataSources delivered with SAP Business Content and DataSources delivered by partners are included in the **D version** (D for delivered). To work with these DataSources, you have to transfer them from the Business Content. When you perform the transfer, the contents of the D version are copied and transferred to an **A version** (A for active). The corresponding structures, tables and programs that are used for the extraction are generated by the system.

After the transfer has taken place, you can adapt the DataSources (enhance the extraction structure, for example). This is possible since it is not the D version of the DataSources but the A version derived from them that is changed.

The transfer is made with the transaction *SBIW* in the SAP source system.

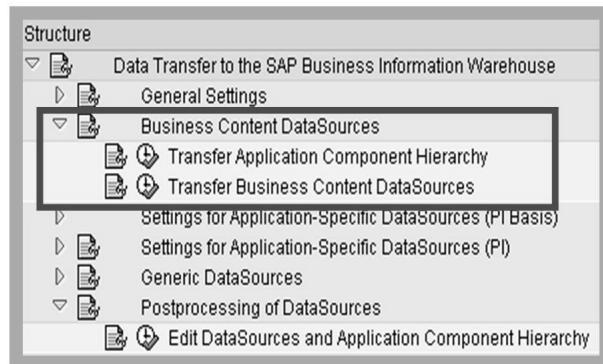


Figure 66: Transfer of Business Content DataSources

The process is divided into two parts: The transfer of the application component hierarchy and the transfer of the DataSources.

The application component hierarchy organizes all the Business Content DataSources into a hierarchical tree, similar to the application component hierarchy in BI. You have to transfer this hierarchy from the Business Content so that the DataSources can be arranged properly in the hierarchy when they are transferred at a later stage. You can only transfer the application component hierarchy as a whole. Selective transfer is not possible. You can, however, adapt the hierarchy at a later stage (see the above graphic: '*Editing DataSources and the Application Component Hierarchy*').

DataSources can be transferred selectively in one of two ways.

1. To transfer and activate a DataSource delivered by SAP with Business Content, in transaction SBIW in the source system, choose *Business Information Warehouse* → *Business Content DataSources* or *Activating SAP Business Content* → *Transfer Business Content DataSources* .
2. When activating BI Content in BI, you can also activate DataSources remotely from the BI system. This activation is subject to an authorization check. You need role *SAP_RO_BCTRA*. Authorization object *S_RO_BCTRA* is checked. The authorization is valid for all DataSources of a source system.



Caution: Remote activation is not supported in source systems and/or in BI systems with a BI Service API release lower than SAP NetWeaver 2004s. In this case, you have to activate the DataSources in the source system manually and then replicate them to the BI system.

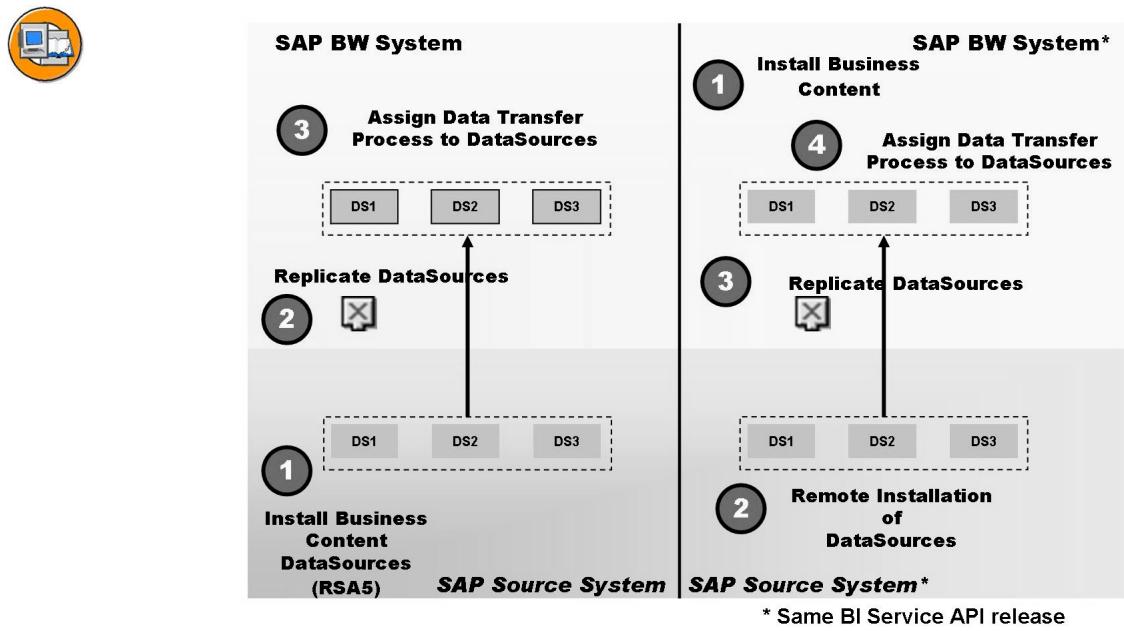


Figure 67: Using Business Content DataSources

→ **Note:** The process of transferring BI Business Content objects (InfoSources, queries, InfoCube, and so on) is described in the first unit of the lesson '*BI Content*'.



Lesson Summary

You should now be able to:

- Describe the transfer process for Business Content DataSources
- Describe how the system decides which DataSource type has to be generated

Lesson: Logistics Data Extraction

Lesson Overview

This lesson describes how data is extracted with the DataSources provided with the Business Content from the Logistics application.



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the necessary steps to extract data from the Logistics application
- Explain the functions in the Logistics Extraction Structure Customizing Cockpit
- Explain the role played by the restructuring (setup) tables and the delta queue in Logistics data extraction
- Describe the update methods to fill the delta queue

Business Example

Your company would like to set up reporting using sales and purchasing data from your SAP source system in BI. To do this, you want to use the logistics DataSources supplied with the Business Content.

Logistics Data Extraction

The following graphic illustrates the data flow for the logistics data extraction by means of an application example from sales:

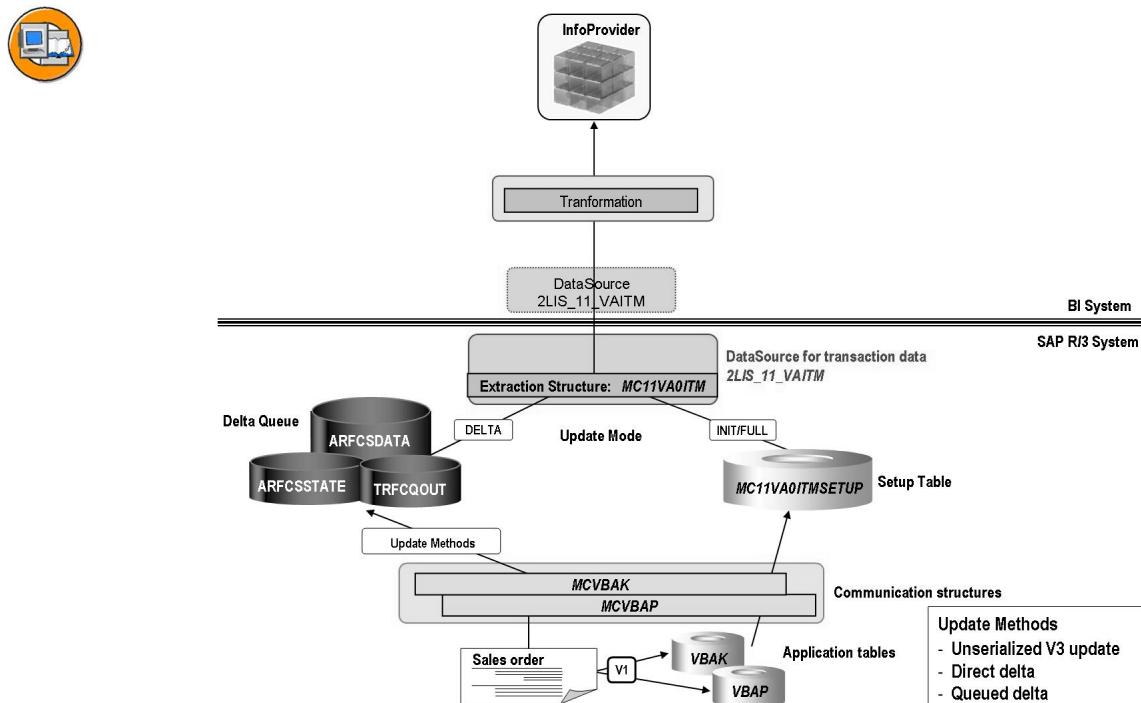


Figure 68: Data Flow: Logistics Data Extraction

The extraction structure and DataSource are maintained in the Logistics Extraction Structures Customizing Cockpit in the transaction *SBIW*:

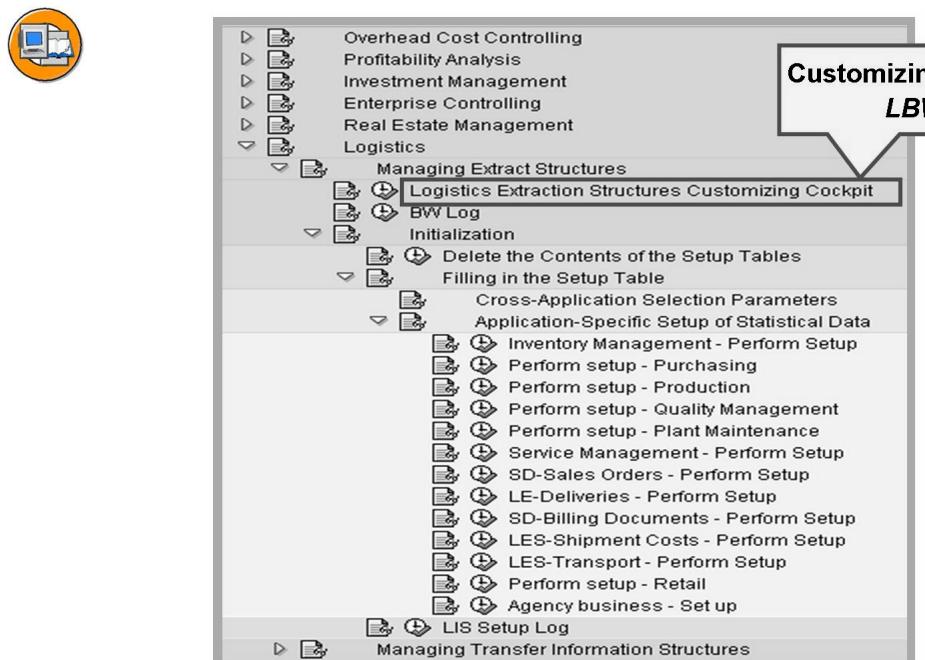


Figure 69: SBIW: Logistics Data Extraction

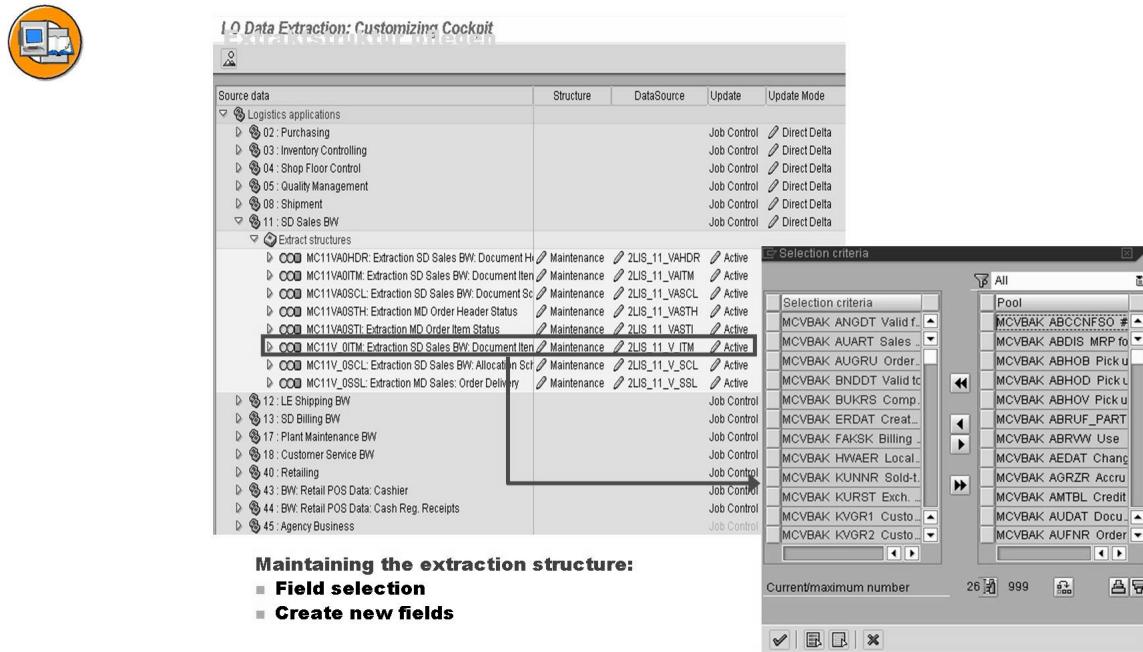


Figure 70: Maintaining extraction structure

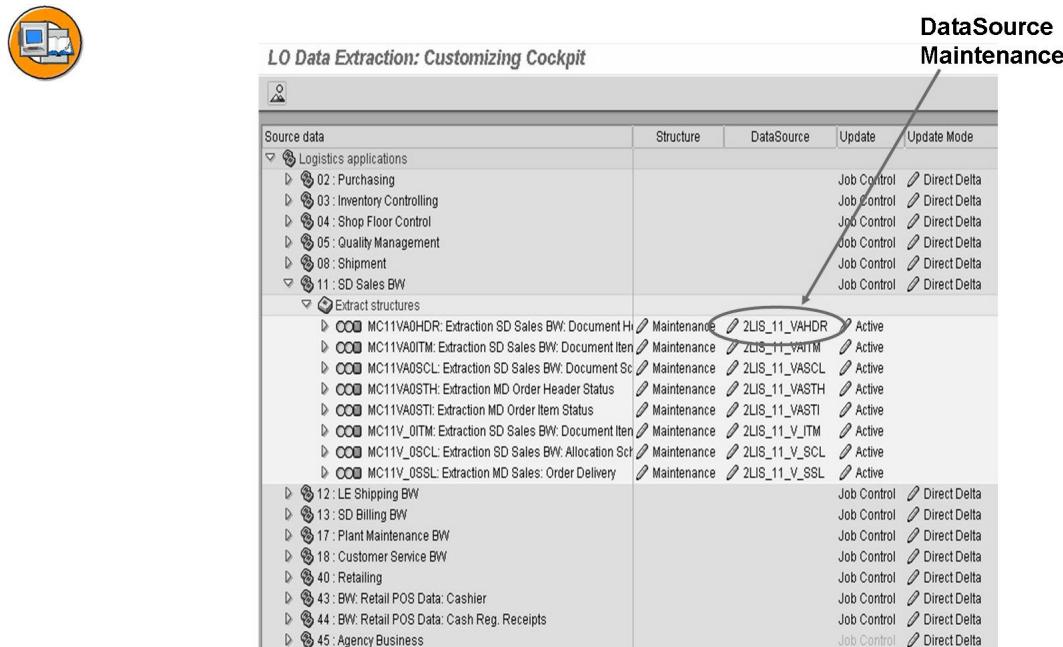


Figure 71: Maintaining DataSources (1)

- Logistics Extraction Structure Customizing Cockpit (LBWE)

The Logistics Cockpit contains the following functionality, as specified in the procedure sequence:

1. Maintaining extraction structures

Every extraction structure can be maintained by you and by SAP. The extraction structure is filled with the assigned communication structures. You can only use selected fields from the communication structures. SAP already delivers extraction structures. You can enhance them as well. After you set up the extraction structure, the system automatically generates it. This completes missing fields (associated units and compounded characteristics). The extraction structure is created hierarchically according to the communication structures. Every communication structure leads to the generation of a substructure that belongs to the actual extraction structure.

2. Maintenance of DataSources

At this point, you call the general maintenance of DataSources, where you can set up the selection of selectable fields and the ability of fields to be negated.

3. Activation of updating

When you activate, data is written to the extraction structures immediately - both online and when you fill the tables for restructuring.

4. Job control

5. Update mode

Here you can set which type of accrued data is to be updated during delta updates. The various update methods are described in more detail later in the lesson.

- Initialization/Setup

The initialization is set up from the OLTP side. A restructuring fills restructuring tables that are then read during initialization. In order to enable the restructuring to be restarted after termination, assign a name to each background run of a restructuring. If a restructuring terminates then or when a restructuring from archived documents is to be terminated, the status of the restructuring is saved under this run name. When there is a restart with the same name, you continue from the saved intermediate status. After a run has terminated successfully, the saved intermediate status is deleted.

The restructuring should run in the background.

- Deleting content from the restructuring tables

Since the restructuring tables is not longer needed after initialization, you can delete them again. To be safe, you can also do this before the restructuring in order to guarantee that the data that has not already been structured is available. Deletion of tables is done across clients for performance reasons:

- BW log

With a log, you can check the transfer to *BI*. The log is stored by user and application if the user parameter MCL is set. The last update procedure for each application is recorded; existing log entries are overwritten.

It is recommended that you only use the log for test purposes. You should deactivate it during productive operation.

The extraction structure is filled with the assigned SAP communication structures. You can only use selected fields from the communication structures. SAP internal control fields, for example, are not offered. User appends to the communication structure are available. SAP already delivers extraction structures, which you can enhance (by connecting to the communication structure). Every extraction structure can be maintained by you and by SAP.

After you set up the extraction structure, the system automatically generates it. This completes missing fields (their units and characteristics). The extraction structure is created hierarchically according to the SAP communication structures. Every communication structure leads to the generation of a substructure that belongs to the actual extraction structure.

At this point, call up the general maintenance screen for DataSources by selecting the extract structure fields that you want to transfer into *BI*. This selection forms the basis for the generating process of the DataSource.

There is a DataSource (for example “2LIS_11_VAITM”) for each extraction structure that is made available (for example, “MC11VA0ITM”) in the OLTP system. A maintenance screen is displayed in which you can assign other properties to the fields of the extraction structure.

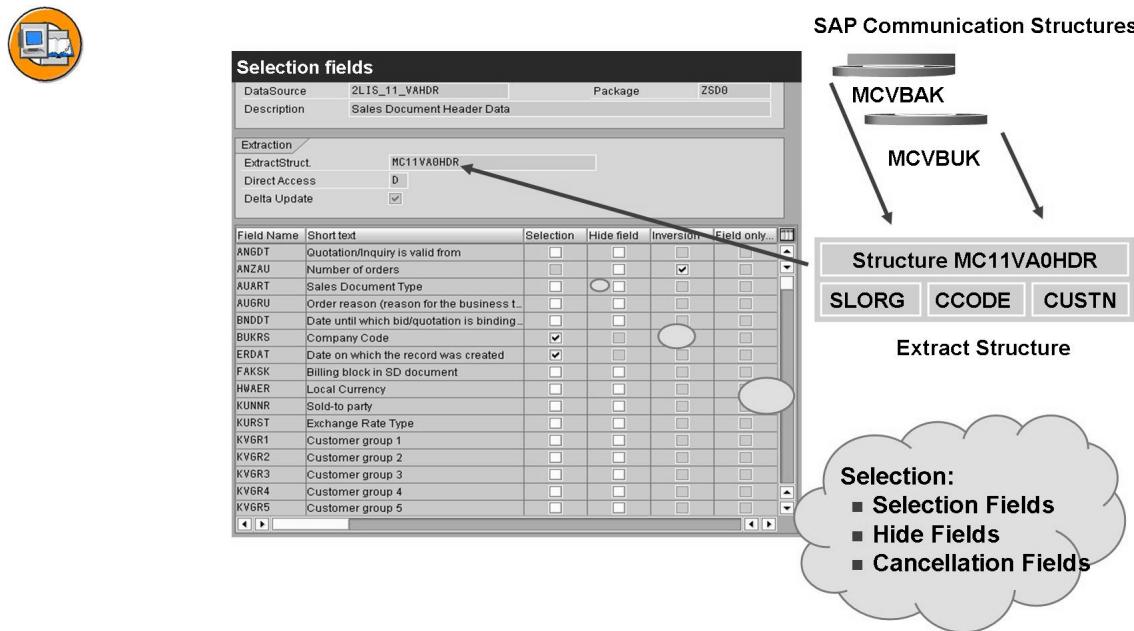


Figure 72: Maintaining DataSources (2)

- Selection fields
- Hide fields
- Cancellation fields (field is inverted when canceled (*-1))

Fields for the DataSource for the item level data will include some fields from the header level, to ensure that all the important data is available.

The DataSource must be replicated in *BI* before it can be used in *BI*. To do this, choose the Source Systems tab page in the BI Data Warehousing Workbench. Position the cursor on the desired source system and use the context menu to display the *Display DataSource Tree*. Navigate to the application components for your DataSource and use the context menu to *Replicate Metadata*. All of the DataSources for this application component from this source system that are not yet present in *BI* are transferred to *BI*.

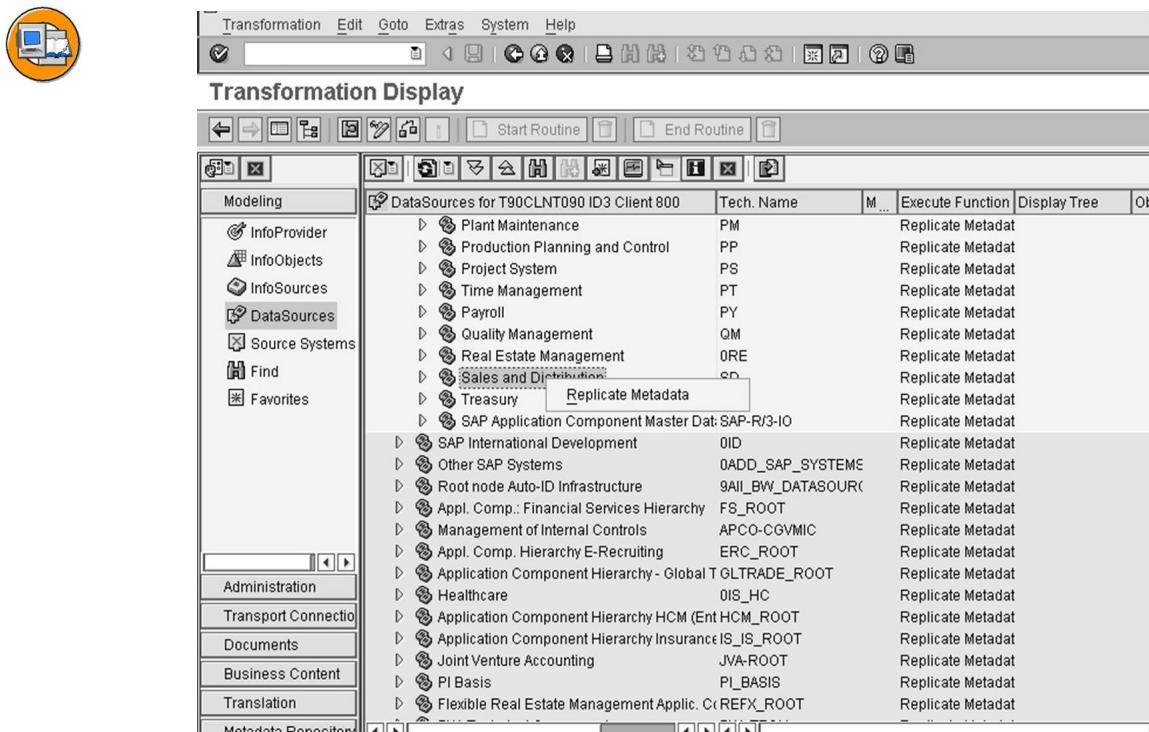


Figure 73: Replicating Metadata

**Hint:**

1. As an alternative, you can also replicate the DataSources for an entire source system. To do this, position the cursor on a source system and choose *Replicate Metadata* from the context menu. However, this replicates all DataSources from this source system, which can take a very long time, and is thus not recommended.
2. If a DataSource is replicated a second time (for example, because it changed in the source system), only the affected DataSource can be replicated as well. To do this, position the cursor on the associated DataSource and choose *Replicate Metadata* from the context menu.
3. If DataSources have changed in the source system (for example, when your SAP System is upgraded), you can decide if you want to activate them to the “A version”, or continue using the existing ones.

When you activate the update, data is then written to the extraction structure, both online and when filling the restructuring tables. It is important that no one generates or changes application data during the initialization phase, at least until the defined tables can be structured. Using selective initializations of the Delta Process the specific window of time needed to suspend application loading can be minimized.



Activate the extraction structures					
Source data	Structure	DataSource	Update	Update Mode	
Logistics applications					
02 : Purchasing			Job Control	<input type="checkbox"/> Direct Delta	
03 : Inventory Controlling			Job Control	<input type="checkbox"/> Direct Delta	
04 : Shop Floor Control			Job Control	<input type="checkbox"/> Direct Delta	
05 : Quality Management			Job Control	<input type="checkbox"/> Direct Delta	
08 : Shipment			Job Control	<input type="checkbox"/> Direct Delta	
11 : SD Sales BW			Job Control	<input type="checkbox"/> Direct Delta	
Extract structures			Job Control	<input type="checkbox"/> Direct Delta	
MC11VA0HDR: Extraction SD Sales BW: Document Header	Maintenance	2LIS_11_VAHDR	<input checked="" type="checkbox"/> Active		
MC11VA0ITM: Extraction SD Sales BW: Document Item	Maintenance	2LIS_11_VAITM	<input checked="" type="checkbox"/> Active		
MC11VA0SCL: Extraction SD Sales BW: Document Structure	Maintenance	2LIS_11_VASCL	<input checked="" type="checkbox"/> Active		
MC11VA0STH: Extraction MD Order Header Status	Maintenance	2LIS_11_VASTH	<input checked="" type="checkbox"/> Active		
MC11V_0ITM: Extraction SD Sales BW: Document Item	Maintenance	2LIS_11_V_ITM	<input checked="" type="checkbox"/> Active		
MC11V_0SCL: Extraction SD Sales BW: Allocation Structure	Maintenance	2LIS_11_V_SCL	<input checked="" type="checkbox"/> Active		
MC11V_0SSL: Extraction MD Sales: Order Delivery	Maintenance	2LIS_11_V_SSL	<input checked="" type="checkbox"/> Active		
12 : LE Shipping BW			Job Control	<input type="checkbox"/> Direct Delta	
13 : SD Billing BW			Job Control	<input type="checkbox"/> Direct Delta	
17 : Plant Maintenance BW			Job Control	<input type="checkbox"/> Direct Delta	
18 : Customer Service BW			Job Control	<input type="checkbox"/> Direct Delta	
40 : Retailing			Job Control	<input type="checkbox"/> Direct Delta	
43 : BW: Retail POS Data: Cashier			Job Control	<input type="checkbox"/> Direct Delta	
44 : BW: Retail POS Data: Cash Reg. Receipts			Job Control	<input type="checkbox"/> Direct Delta	
45 : Agency Business			Job Control	<input type="checkbox"/> Direct Delta	

Figure 74: Activating the Update

When you extract transaction data with the LO method, you need a so-called restructuring, which is similar in its usage to the LIS method for refreshing statistical data. The restructuring reads the dataset you want to edit (for example, customer orders with the tables VBAK, VBAP and so on) and fills the relevant communication structure. The data is stored in cluster tables (<Extraction structure>_SETUP), from where they are read during the initialization run.



Statistical Setup from Old Documents: Orders													
1. Selection criteria to limit historical data.													
Document data restriction													
Archiving session													
Sales Organization													
Company code													
Sales document													
Control of the setup run													
Name of run	<input checked="" type="checkbox"/>												
New run	<input checked="" type="checkbox"/>												
Termination date	01.04.2005												
Termination time	13:24:44												
Block all orders?	<input type="checkbox"/>												
No. tolerated faulty documents													
Extraction structures BW	<input checked="" type="checkbox"/>												
Simulation extr. str. BW	<input type="checkbox"/>												
3. Name of run													
Simulation:	No update!												
4. New Run Indicator													
LWBF:	<table border="1"> <thead> <tr> <th>Field name</th> <th>DataSource</th> </tr> </thead> <tbody> <tr> <td>MC11VA0HDR</td> <td>2LIS_11_VAHDR</td> </tr> <tr> <td>MC11VA0ITM</td> <td>2LIS_11_VAITM</td> </tr> <tr> <td>MC11VA0SCL</td> <td>2LIS_11_VASCL</td> </tr> <tr> <td>MC11V_0ITM</td> <td>2LIS_11_V_ITM</td> </tr> <tr> <td>MC11V_0SCL</td> <td>2LIS_11_V_SCL</td> </tr> </tbody> </table>	Field name	DataSource	MC11VA0HDR	2LIS_11_VAHDR	MC11VA0ITM	2LIS_11_VAITM	MC11VA0SCL	2LIS_11_VASCL	MC11V_0ITM	2LIS_11_V_ITM	MC11V_0SCL	2LIS_11_V_SCL
Field name	DataSource												
MC11VA0HDR	2LIS_11_VAHDR												
MC11VA0ITM	2LIS_11_VAITM												
MC11VA0SCL	2LIS_11_VASCL												
MC11V_0ITM	2LIS_11_V_ITM												
MC11V_0SCL	2LIS_11_V_SCL												

Figure 75: Initialization/Simulation (OLI*BW)

A full update into *BI* is also performed with the restructuring tables.

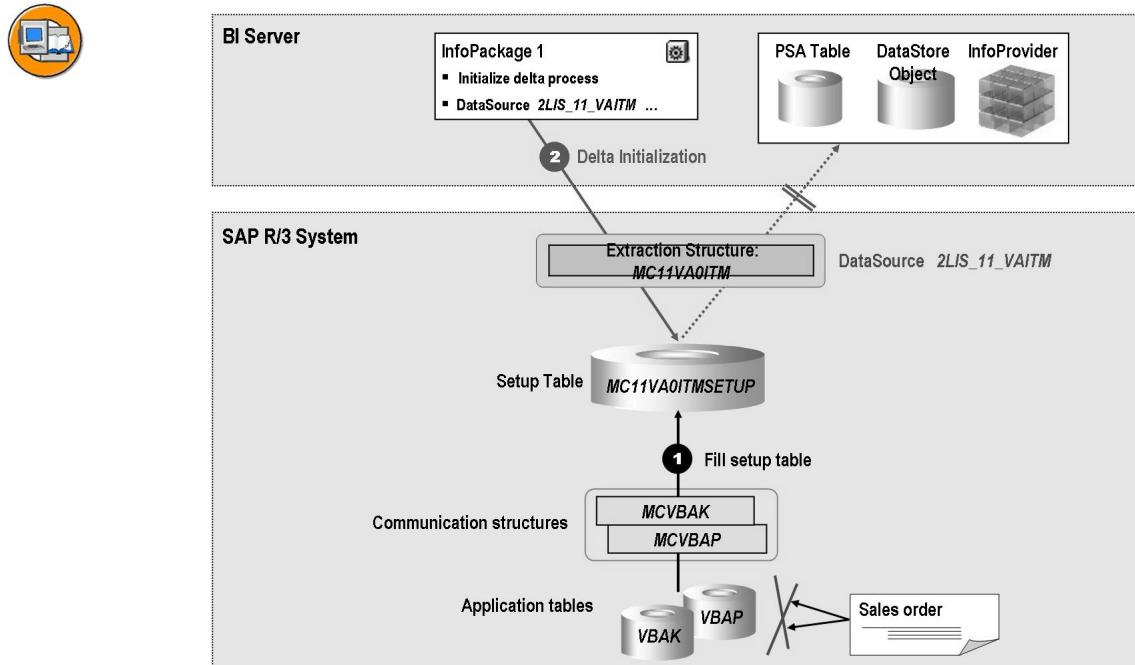


Figure 76: Filling the Restructuring Table / Delta Initialization

You have to initialize (“build”) the dataset of the prior X number of years of transactions, for the respective extraction structures, before data can be extracted from the SAP system. The initialization is set up in the OLTP. The restructuring process fills restructuring tables that are read during the initialization.

→ **Note:** The initialization of the transactions dataset can also be built upon organizational elements such as, Sales Organization or Company Code.

It is recommended that you run this in simulation mode beforehand (the simulation mode is only possible in some application areas: SD customer sales orders, LE deliveries, SD billing documents). You have the option of checking the extraction structure update, without actually updating the data. The results are recorded in a detailed simulation log.

If you want to transfer data in the delta process, the process must be initialized in the first data request. In this process, the selection conditions for connecting delta uploads are set and its complete dataset is loaded into *BI* from the SAP system. To do this, you must select the radio buttons “Initialize Delta Process” and “Initialization with Data Transfer” for the InfoPackage under the update parameters of the Scheduler.

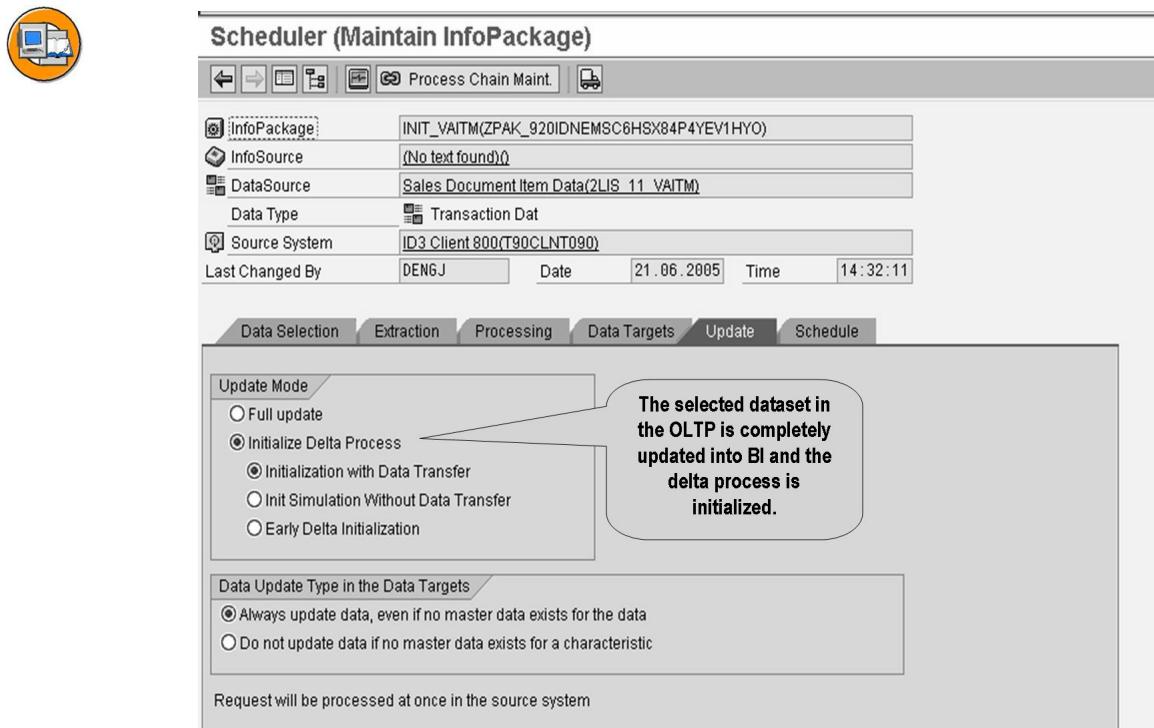


Figure 77: InfoPackage: Delta Initialization

When restructuring statistics data in the *SAP* source system, data can be read from the tables for the application (resident documents) or from archived documents. Some aspects of the extraction of archive files from *SAP* source systems will be considered in the following:

Archive File

Here you specify the name of the archive from which archived documents are to be read. If you do not specify an archive, the documents are read from the residence (the documents found in the system).

Archiving Run

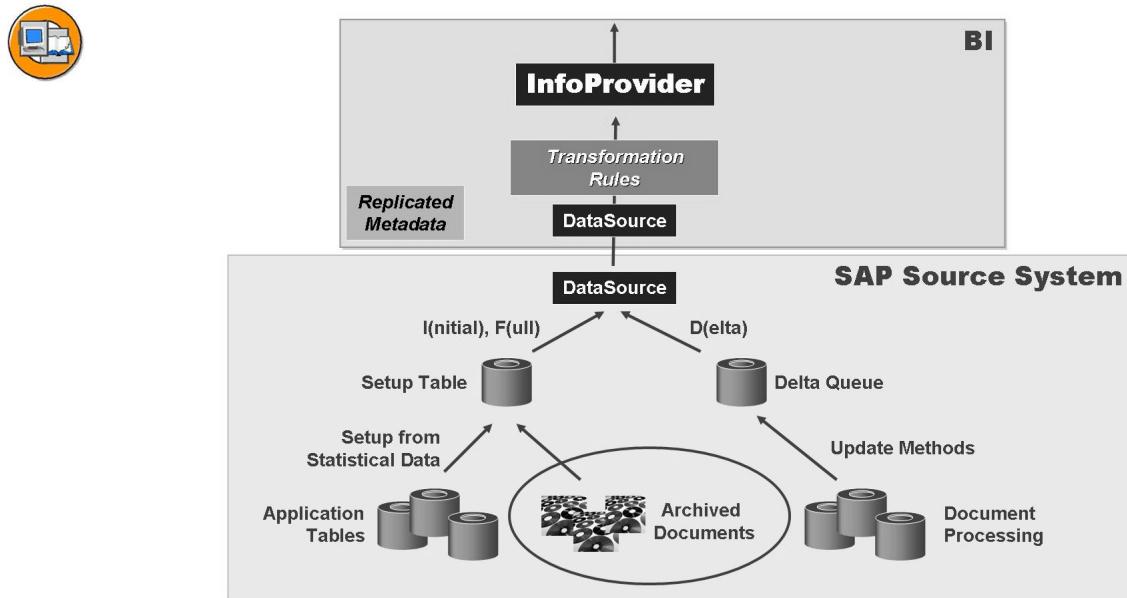


Figure 78: Extracting Archived Data

Archiving runs have sequential numbers that are automatically counted up. The numbers for archiving runs are always valid across objects.

Please note the following:

When restructuring from statistics data, the setup tables for an application are filled for the DataSources that have an active update status. Transfer of data from these setup tables is only possible in update mode (Initialization of the delta process) and (Delta update). If data is loaded into *BI* from the setup table of an application with full update, the delta process is not affected by this. However, you need to be sure that records that were already transferred are not sent to *BI* again (especially during later update to an InfoCube). It is recommended that you load archived documents into the data targets of *BI* before initializing a delta process by full update. If the DataSources are already actively working in a delta process and are writing to an DataStore object as an InfoProvider, a full update is only possible when the “Repair Full Request” indicator is set in the InfoPackage menu.

Please note that this functionality is only possible in some logistics applications (such as MM and SD). If the functionality exists, you will see a selection field for archiving runs in the application-specific restructuring of this application (SBIW).

These methods determine the type of accrued data that is updated during delta postings. The following update methods are offered:



You can select the update methods from the logistics extraction structures Customizing Cockpit (LBWE) for each application:

LO Data Extraction: Customizing Cockpit				
Source data	Structure	DataSource	Update	Update Mode
Logistics applications				
02 : Purchasing			Job Control	<input type="radio"/> Direct Delta
03 : Inventory Controlling			Job Control	<input type="radio"/> Direct Delta
04 : Shop Floor Control			Job Control	<input type="radio"/> Direct Delta
05 : Quality Management			Job Control	<input type="radio"/> Direct Delta
08 : Shipment			Job Control	<input type="radio"/> Direct Delta
11 : SD Sales BW			Job Control	<input checked="" type="radio"/> Direct Delta
Extract structures				
MC11VA0HDR: Extraction SD Sales BW: Document Header	Maintenance	2LIS_11_VAHDR	Active	
MC11VA0ITM: Extraction SD Sales Item	Select Update Mode			
MC11VA0SCL: Extraction SD Sales Line				
MC11VA0STH: Extraction MD Order Header				
MC11VA0STI: Extraction MD Order Item				
MC11V_0ITM: Extraction SD Sales Item				
MC11V_0SCL: Extraction SD Sales Line				
MC11V_0SSL: Extraction MD Order Item				
LE Shipping BW				
SD Billing BW				
Plant Maintenance BW				
Customer Service BW				
Retailing				
BW: Retail POS Data: Cashier				
BW: Retail POS Data: Cash Reg. Receipts				
Agency Business				
12 : LE Shipping BW			Job Control	<input type="radio"/> Direct Delta
13 : SD Billing BW			Job Control	<input type="radio"/> Direct Delta
17 : Plant Maintenance BW			Job Control	<input type="radio"/> Direct Delta
18 : Customer Service BW			Job Control	<input type="radio"/> Direct Delta
40 : Retailing			Job Control	<input type="radio"/> Direct Delta
43 : BW: Retail POS Data: Cashier			Job Control	<input type="radio"/> Direct Delta
44 : BW: Retail POS Data: Cash Reg. Receipts			Job Control	<input type="radio"/> Direct Delta
45 : Agency Business			Job Control	<input type="radio"/> Direct Delta

Figure 79: Update methods

Unserialized V3 update:

With this update mode, the extraction data of the application being viewed is written to the update tables with an V3 update module and is kept there until the data is read with an update collection run and processed. The data in the update collection run, however, is read from the update table without considering sequence and is transferred to the delta queue.

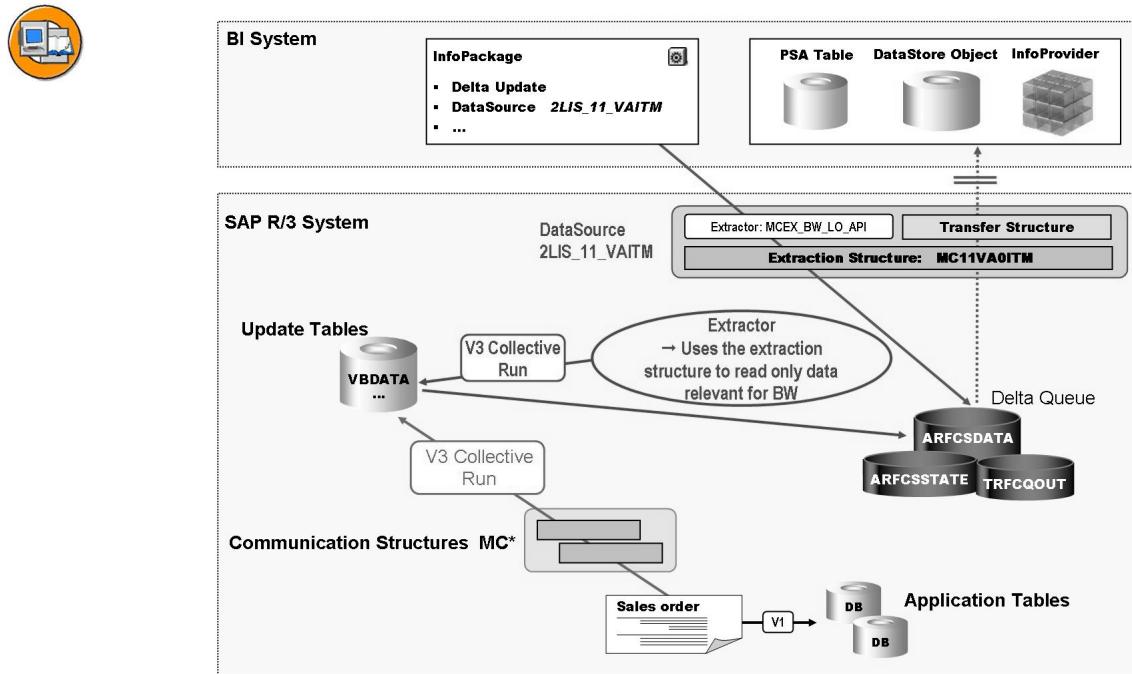


Figure 80: Unserialized V3 Update

The method “Unserialized V3 Update” does not ensure serialization of the document data. Update to the DataStore objects is not recommended if serialization is desired (for example, for reporting purposes)

Direct delta:

With this update mode, the extraction data is transferred directly to the delta queue with every document posting. The sequence of the transfer thus agrees with the chronological order of data creation.

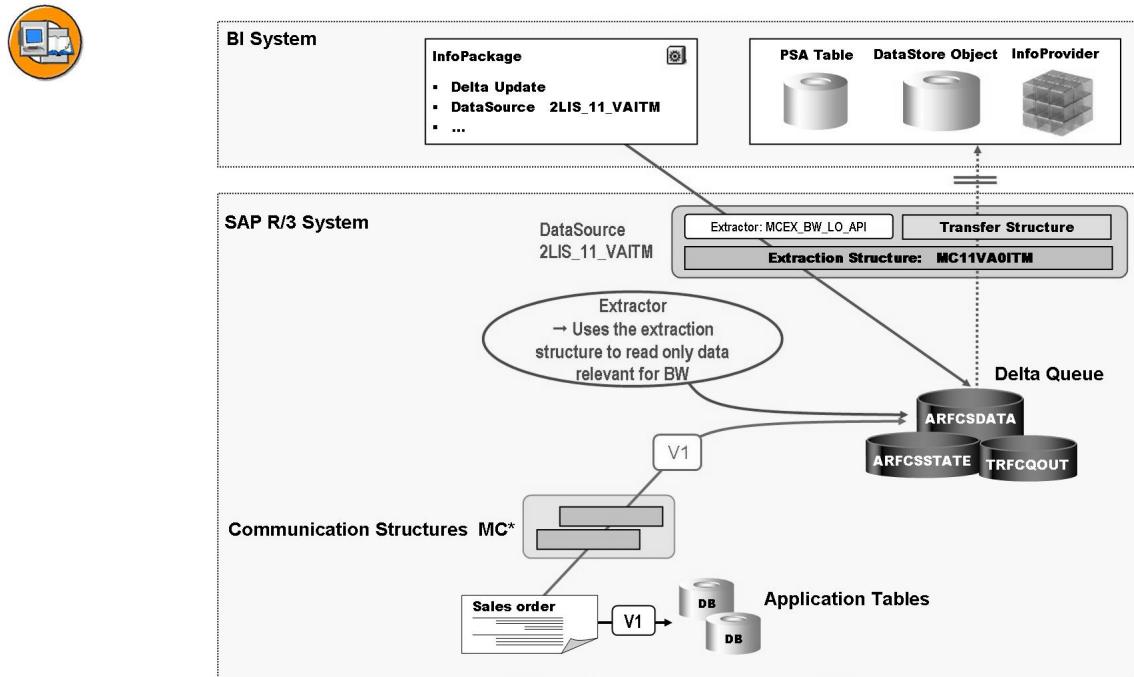


Figure 81: Direct delta

Benefits and properties of “Direct Delta”:

- By writing to the delta queue within the V1 posting process, serialization by document is ensured with the enqueue concept of applications.
- For customers with fewer documents, the process is recommended if a downtime is possible in the initialization process during restructuring and delta initialization requests.
- V1 is more burdened by this process than with V3 or “Queued Delta”. For customers with the amount of documents mentioned above, this is not really a factor.
- Extraction is independent of V2 updating.
- Additional monitoring of update data or extraction queue is not required.

Queued delta:

With this update mode, instead of being collected in the update data, the extraction data from the affected application is collected in an *extraction queue* and can be transferred to the delta queue, in a similar manner to the V3 update, with an update collection run. Depending on the application, up to 10,000 delta extractions of documents can be aggregated to an LUW in the delta queue per DataSource.

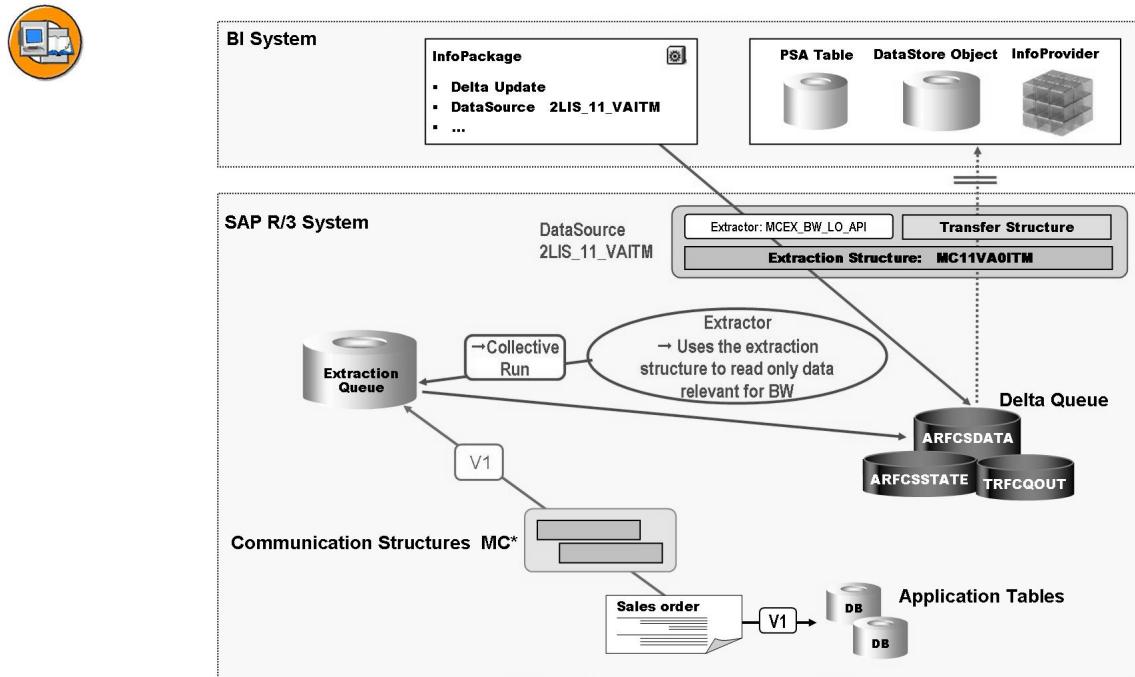


Figure 82: Queued delta

The logistics queue overview function (transaction LBWQ) is available for monitoring the extraction queue.

Benefits and properties of “Queued Delta”:

- By writing to the extraction queue in the V1 update process, serialization by document is ensured with the enqueue concept for applications.
- Due to the collection of data in the extraction queue that is regularly processed (SAP recommends hourly), the process is especially recommended for customers with a high amount of documents.
- The collection run uses the same reports as previously (RMBWV311,...).
- In the initialization process, the collection of new document data during the delta initialization request can reduce the downtime on the restructuring run (filling the setup tables).
- V1 is much more difficult than the use of V3.
- In contrast to the V3 collection run, a definitive end of the collection run is within reach and a subsequent process can be scheduled. After the collection run for an application has finished, an event (&MCEX_11,...) is triggered automatically, that – can be used – at the start of a subsequent job.
- Extraction is independent of V2 updating.



Hint: As of PI 2002.1, there were changes to the update methods in the logistics extraction (for more information, see SAP Note 505700).

Examples of the individual steps of logistics data extraction and procedures are provided as an overview in the following:

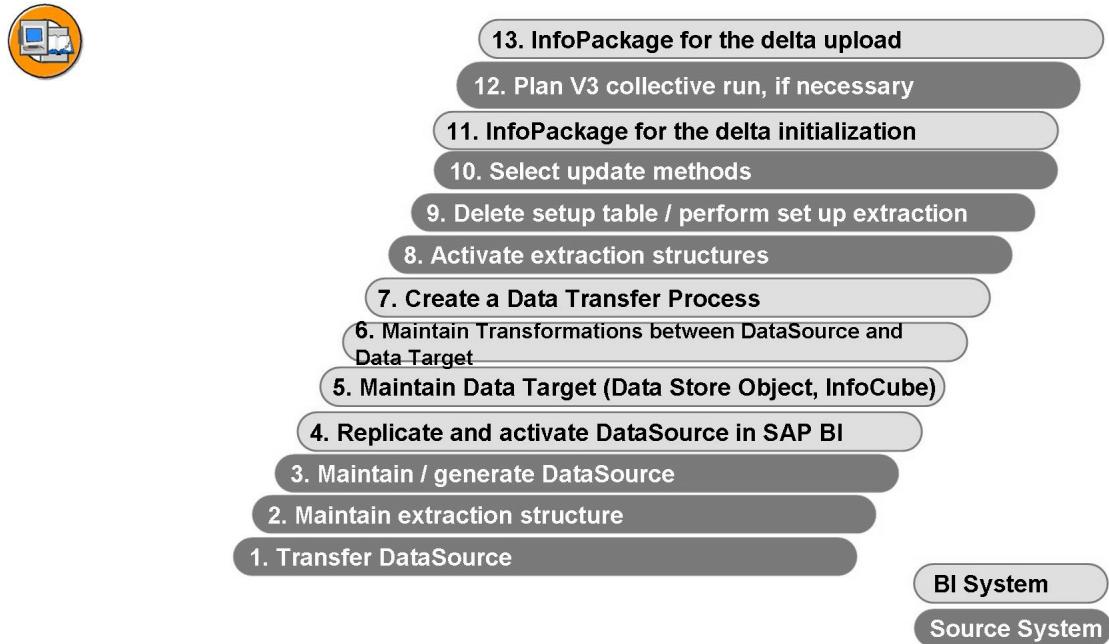


Figure 83: LO Data Extraction Procedure

The following section describes the steps that are necessary to configure a data flow for logistics data extraction.

1. Transfer the logistics DataSource for transaction data from Business Content

With the transfer of the DataSource, the associated extraction structure is also delivered, but the extraction structure is based on LIS communication structures. Furthermore, based on the extraction structure for the DataSource, a restructuring table that is used for the initialization and full update to BI is generated.

Naming convention:

DataSource 2LIS_<Application>_<Event><Suffix>; where <Event> is optional

Examples:

2LIS_11_VAITM: 11 = SD Sales VA = sales order, delivery schedule ITM
(\rightarrow ITEM) = document position

2LIS_02_HDR: 02 = MM purchasing HDR (\rightarrow HEADER) = Document header

Extraction structure MC<Application><Event/group of events>0<Suffix>; where MC is derived from the associated communication structures and <Suffix> is optional

Examples:

MC1IVA0ITM: Extraction structure for the DataSource 2LIS_11_VAITM

MC02M_0HDR: Extraction structure for the DataSource 2LIS_02_HDR, where M_ indicates the group for the events MA (order), MD (delivery schedule), ME (contact) and MF (request).

Restructuring table (= setup table) <Extraction structure>SETUP

Example:

Extraction structure: *MC1IVA0IT* \Rightarrow Restructuring table:

MC1IVA0ITSETUP

2. Maintain extraction structure (transaction LBWE)

This means that fields can be added to the extraction structure that is delivered with the DataSource without modifying anything. On the one hand, fields from the LIS communication structures that are assigned to the extraction structure can be used, that means standard fields that SAP has not selected, and on the other hand, customer fields that were attached to the LIS communication structures with the append technique can be used. After the extraction structure is created, it is generated automatically and the associated restructuring table is adapted.

3. Maintain/generate DataSource

In the DataSource maintenance, you can assign the properties *Selection*, *Hide*, *Inversion* (= *Cancellation*) and *Field Only Known in Customer Exit* to the fields of the extraction structure. After enhancing the extraction structure, the DataSource always has to be generated again!

4. Replicate and activate DataSource in SAP BI (=metadata upload)

5. Maintain Data Target (Data Store Object, InfoCube)

6. Maintain Transformations between DataSource and Data Target

7. Create a Data Transfer Process

the Data Transfer Process will be used later to update the data from the PSA table into the Data Target.

8. Set extraction structure for updating to active (transaction LBWE)

In this way, data can be written to the restructuring table or the delta queue from then on using the extraction structure (see following steps).

9. Filling the restructuring table/restructure (OLI*BW)

During this process, no documents should be created or changed in the system! In some applications, it is possible to fill the restructuring table beforehand in simulation mode. These results are listed in a log (transaction *LBWF*). Before filling the restructuring table, you must ensure that the content of the tables is deleted (transaction *LBWG*), preventing the table from being filled multiple times. Once the restructuring tables are filled, document editing can resume as long as *Unserialized V3 Update* or *Queued Delta* is selected in the next step. Be absolutely sure that no V3 collection run is started until the next successful posting of an InfoPackage for delta initialization (see step 15).

10. Select update method
 - Unserialized V3 update
 - Queued delta
 - Direct delta
11. Create an InfoPackage for the DataSource and schedule the *Delta Initialization* in the Scheduler

This updates the BI-relevant data from the restructuring table to the PSA table. Since the restructuring table is no longer needed after delta initialization, the content can be deleted (transaction *LBWG*).

Use the Data Transfer Process created in step 7 to update the data from the PSA table into the Data Targets. After successful delta initialization, document editing can resume, as long as the direct delta update method was selected in step 13. This means that BI-relevant Δ data is written directly to the delta queue.

Note:

If the DataSource supports **early-delta initialization**, the Δ data can be written to the delta queue during delta initialization. This feature is controlled with an indicator in the Scheduler.

12. Start V3 collection run (transaction *LBWE*)

This step is only necessary when the update method *unserialized V3 Update* or *Queued Delta* was selected in step 13. By starting a corresponding job for an application, the BI-relevant Δ data is read from the update tables or extraction queue and written to the delta queue.
13. Create an InfoPackage for the DataSource in BI and schedule the *Delta Update* in the Scheduler

The BI-relevant Δ data from the delta queue for the DataSource is updated to the PSA table. Use the Data Transfer Process created in step 7 to update the data from the PSA table into the Data Targets.

Exercise 5: Overview of the LO Extraction (LO Cockpit)

Exercise Objectives

After completing this exercise, you will be able to:

- Activate the LO Business Content extractors

Business Example

Your company is implementing the LO extractors and phasing out the LIS extraction method. It is your task to check and activate the LO extractors.



Hint: In these exercises, the transport status of all the data sources could be different, as mentioned in the general notes about the exercises at the beginning of this course. Dialog boxes for the transport may be displayed for these data sources. If you are not sure what to enter in these dialog boxes, ask the instructor! In many cases a new transport request will be required and/or the development class “Z001”.

In the case of a small number of the extractors below, you are not allowed to change the extraction structure either. Ignore this error by pressing Enter. This exercise concentrates mainly on menu paths.

Task 1:

You will be given an LO DataSource from the following list. Each group number in the list is assigned to a DataSource.

Group number	Application component / DataSource
01 / 21	02 / 2LIS_02_ITM
02 / 22	03 / 2LIS_03_BF
03 / 23	04 / 2LIS_04_PEARBPL
04 / 24	05 / 2LIS_05_Q0ITEM
05 / 25	08 / 2LIS_08TRFKP
06 / 26	11 / 2LIS_11_VAHDR
07 / 27	12 / 2LIS_12_VCITM
08 / 28	13 / 2LIS_13_VDITM
09 / 29	17 / 2LIS_17_IOITEM
10 / 30	18 / 2LIS_18_IOITEM
11 / 31	40 / 2LIS_40_REVAL
12 / 32	02 / 2LIS_02_HDR
13 / 33	03 / 2LIS_03_UM
14 / 34	04 / 2LIS_04_PECOMP
15 / 35	05 / 2LIS_05_Q0ACTY
16 / 36	08 / 2LIS_08TRTL
17 / 37	11 / 2LIS_11_VASCL
18 / 38	12 / 2LIS_12_VCHDR

1. Check each step in the LO Extractor Customizing Cockpit for your assigned DataSource.
2. Maintain your assigned extraction structure

Select the first field from the *Pool* column for your DataSource.



Hint: If the *pool* column is empty, skip this step.

3. Maintain your assigned DataSource

Choose the second change icon (pencil) in the “DataSource” column.

This is the standard BI DataSource maintenance screen.

Continued on next page

4. Ensure that your assigned DataSource is “active”.



Hint: If the extraction structure has already been activated, this serves to deactivate it. The status is changed by selecting the icon again.

Task 2:

Check the other critical menu paths in the LO extractor.

DO NOT ACTUALLY EXECUTE THESE TASKS, JUST FOLLOW THE MENU PATHS PROVIDED!

1. Locate the path “Delete the Contents of the Setup tables”. You would do this prior to reloading these tables. The setup tables contain historical data that is used in the delta initialization to BI.

If this were to be done, you would have to enter your application component number next.

2. Locate the path for generating the setup tables for each application area. This is where the setup tables are generated in the SAP system for subsequent loading to BI during the delta initialization.

DO NOT EXECUTE THE PROGRAMS FOR FILLING THE TABLE

Solution 5: Overview of the LO Extraction (LO Cockpit)

Task 1:

You will be given an LO DataSource from the following list. Each group number in the list is assigned to a DataSource.

Group number	Application component / DataSource
01 / 21	02 / 2LIS_02_ITM
02 / 22	03 / 2LIS_03_BF
03 / 23	04 / 2LIS_04_PEARBPL
04 / 24	05 / 2LIS_05_Q0ITEM
05 / 25	08 / 2LIS_08TRFKP
06 / 26	11 / 2LIS_11_VAHDR
07 / 27	12 / 2LIS_12_VCITM
08 / 28	13 / 2LIS_13_VDITM
09 / 29	17 / 2LIS_17_I0ITEM
10 / 30	18 / 2LIS_18_I0ITEM
11 / 31	40 / 2LIS_40_REVAL
12 / 32	02 / 2LIS_02_HDR
13 / 33	03 / 2LIS_03_UM
14 / 34	04 / 2LIS_04_PECOMP
15 / 35	05 / 2LIS_05_Q0ACTY
16 / 36	08 / 2LIS_08TRTLP
17 / 37	11 / 2LIS_11_VASCL
18 / 38	12 / 2LIS_12_VCHDR

1. Check each step in the LO Extractor Customizing Cockpit for your assigned DataSource.
 - a) *OLTP System → Transaction SBIW → Settings for Application Specific Data Sources → Logistics → Managing Extraction Structures → Logistics Extraction Structures Customizing Cockpit*
2. Maintain your assigned extraction structure

Continued on next page

Select the first field from the *Pool* column for your DataSource.



Hint: If the *pool* column is empty, skip this step.

- Expand the tree structure as far as your assigned application component number (above), then go to the “Extraction Structures”.

Select the first change icon (pencil) in the “Structure” column.

Selection criteria	Column “Pool”
“Blue fields”	“Black fields”

Select the first field from the *Pool* column for your DataSource.



Hint: If the *pool* column is empty, skip this step.

Click on the *arrow icon ►*. This will transfer the field to the extraction structure.

Save (Enter)

- Maintain your assigned DataSource

Choose the second change icon (pencil) in the “DataSource” column.

This is the standard BI DataSource maintenance screen.

- Save*

- Ensure that your assigned DataSource is “active”.



Hint: If the extraction structure has already been activated, this serves to deactivate it. The status is changed by selecting the icon again.

- Select the third change icon (pencil) in the “Update” column.

Task 2:

Check the other critical menu paths in the LO extractor.

DO NOT ACTUALLY EXECUTE THESE TASKS, JUST FOLLOW THE MENU PATHS PROVIDED!

- Locate the path “Delete the Contents of the Setup tables”. You would do this prior to reloading these tables. The setup tables contain historical data that is used in the delta initialization to BI.

Continued on next page

If this were to be done, you would have to enter your application component number next.

- a) *OLTP System → Transaction SBIW → Settings for Application Specific Data Sources → Logistics → Managing Extraction Structures → Initialization → Delete the Contents of the Setup Tables*
2. Locate the path for generating the setup tables for each application area. This is where the setup tables are generated in the SAP system for subsequent loading to BI during the delta initialization.

DO NOT EXECUTE THE PROGRAMS FOR FILLING THE TABLE

- a) *OLTP System → Transaction SBIW → Settings for Application-Specific Data Sources → Logistics → Managing Extraction Structures → Initialization → Filling the Setup Tables: Application-Specific Setup of Statistical Data → Choose the relevant application.*



Lesson Summary

You should now be able to:

- Describe the necessary steps to extract data from the Logistics application
- Explain the functions in the Logistics Extraction Structure Customizing Cockpit
- Explain the role played by the restructuring (setup) tables and the delta queue in Logistics data extraction
- Describe the update methods to fill the delta queue

Lesson: Generic Data Acquisition

Lesson Overview

In this lesson, you will get an overview of the options for creating generic DataSources using various extraction methods.



Lesson Objectives

After completing this lesson, you will be able to:

- Create generic DataSources
- Use different data acquisition methods to create generic DataSources
- Describe the emulation process of existing generic and non-generic DataSources

Business Example

Your company wants to analyze data from SAP systems in BI. Since Business Content does not include a DataSource for this purpose, your task now is to generate a DataSource using the tools for generic data extraction.

Generic Data Acquisition

Review the following reasons for using tools for generic data extraction.



When should you use the tools for generic data extraction to create a DataSource?

Business Content does not include a DataSource for your application.

Business content requires additional enhancements that need data that is not supplied by BI.

The application does not feature its own generic data extraction method.

You use your own programs to fill your tables in the SAP system.

Figure 84: Using Tools for Generic Data Extraction

The graphic above will help you decide whether to use tools for generic data extraction.

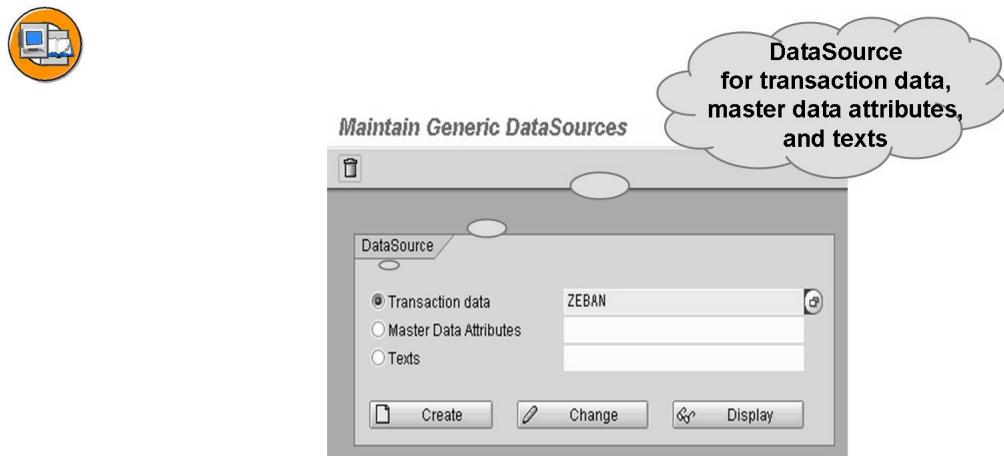


Figure 85: Tools for Generic Data Extraction

You can open the tools for generic data extraction using the implementation guide **Business Information Warehouse** in the SAP source system (transaction SBIW). In the “Generic DataSources” area, you will find the transaction **Maintain Generic DataSources** (transaction RSO2). You can use this transaction to create DataSources for *transaction data*, *master data*, and for *texts*.

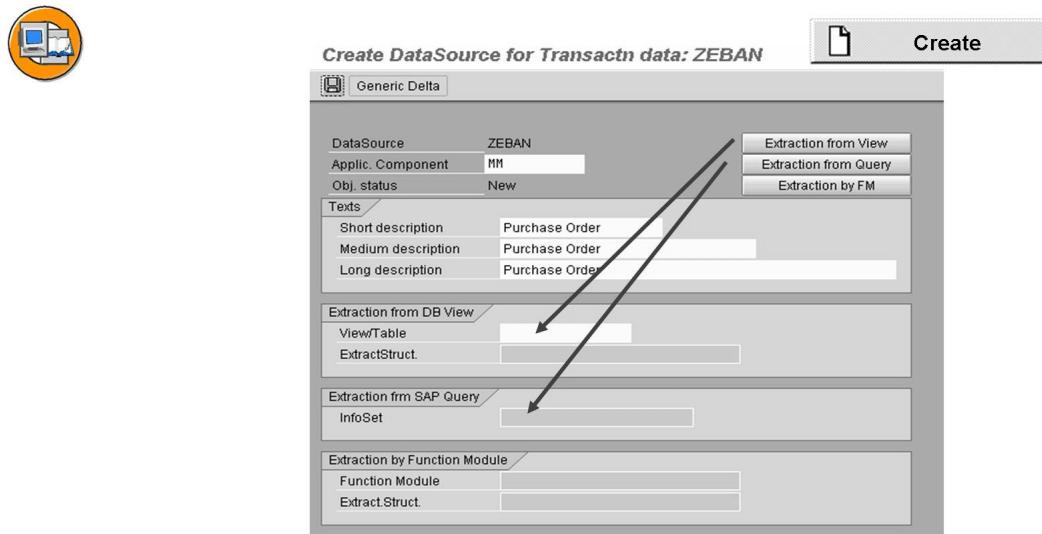


Figure 86: Creating DataSources Without Delta

When you create a generic DataSource you first have to enter the application component and texts for describing the DataSource. Definition of the application component determines where the DataSource metadata is stored after replication of the DataSources in the *Source systems* tab in the Data Warehousing Workbench. You can then select the data source for the generic DataSource. Here you can choose both transparent tables and database views. You can also use SAP Query/ InfoSet Query to extract data. Choose the “Extraction from View” button if you

want to extract the data from a transparent table or database view, and enter the name of the table or view. The extraction structure of the generated DataSource is then identical to that of the view or table. Choose the “Extraction from Query” button if you want to use an SAP Query InfoSet as a data source. Then select the required InfoSet from the InfoSet catalog. You can go to the screen for maintaining an existing InfoSet by double-clicking on its name.

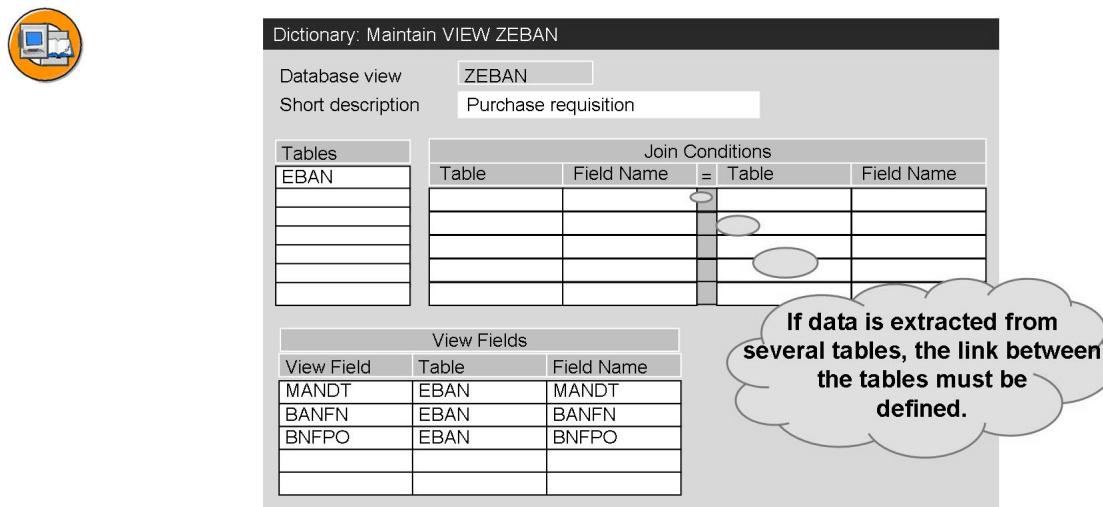


Figure 87: Generic Extraction with View

You create views with transaction SE11 (ABAP Workbench Dictionary). The view type “Database View” must be selected when creating the view. The tables must be joined by identical fields in the join conditions. All of the key fields in the tables must be defined as view fields, otherwise some data records may overlap and the data will not be transferred to BI. Characteristics can be ignored if you are certain that they are not used in the table keys. If characteristics with only one attribute are used, they can be assigned selection conditions (*Goto → Selection Conditions*). If the master data is time dependent, the DATETO and DATEFROM fields have to be included in the view.

The sequence of fields for defining a database view for extracting texts is fixed.

- Client
- Language with the domain SPRAS, if the defined texts are language dependent
- Key field: if compounding is used, all of the relevant characteristics have to be listed.
- If the data is time dependent, the DATETO and DATEFROM fields
- Short, medium and long texts

If one of the text fields is not required, it must not be checked. If there is no medium-length text, the last two fields are the short text and long text fields.

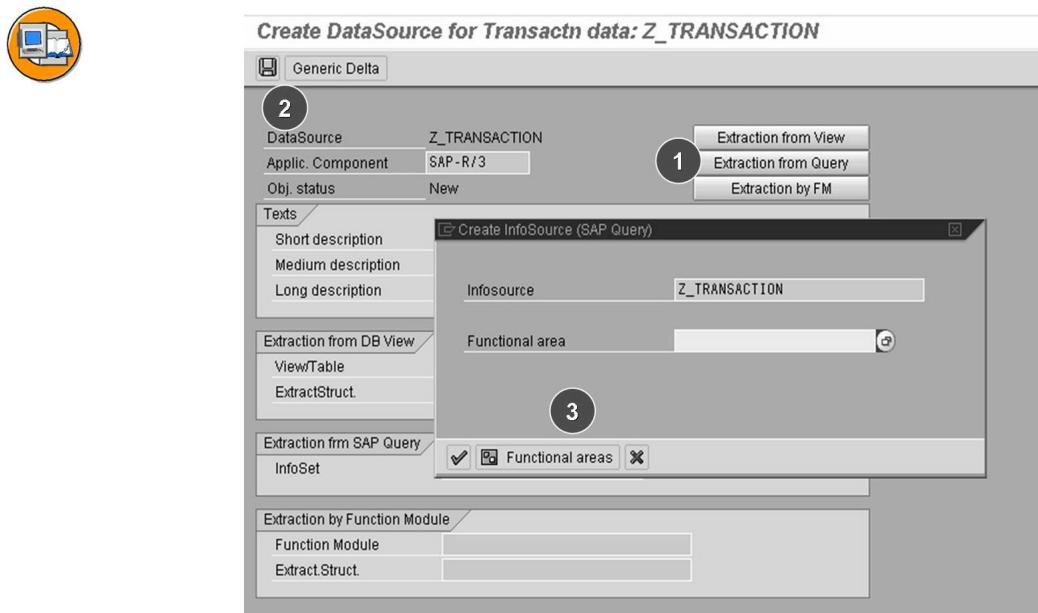


Figure 88: Generic Extraction with InfoSet

SAP Query / InfoSet Query is a powerful tool for defining reports in the SAP Source systems and supports different forms of reporting. It allows users to define and execute their own reports on data in the SAP System without any knowledge of the ABAP programming language.

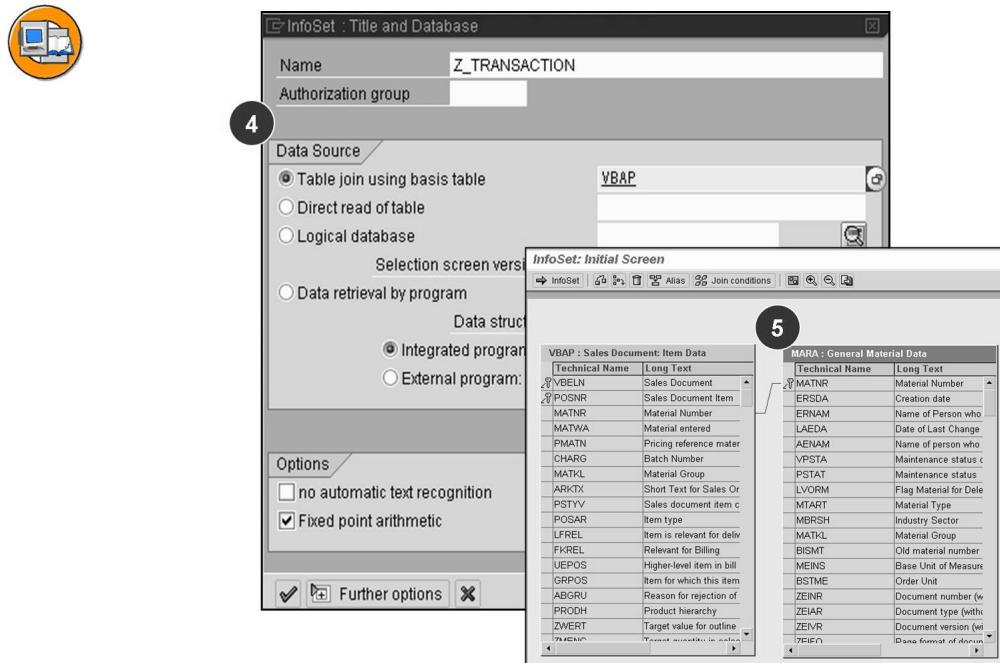
Reports are structured by entering texts and selecting fields and options in SAP Query. The relevant fields can be selected more easily using InfoSets (before Release 4.6C: Functional areas). InfoSets are viewed as “DataSources”. These data sources can be used for extracting into “BI”.

The procedure for creating an InfoSet from the DataSource definition:

1. Fill out all of the required entry fields (application component, descriptions).
2. Choose the *Extraction from Query*.
3. Choose *Save*. A new screen is displayed.
4. Assign a name for your *InfoSet* and choose *InfoSet Maintenance*.
5. In the screen that follows, press *Create* again.

Then carry out the following steps:

- Select the data source (DB tables/views, joins, logical DB,...)
- Define the field groups and assign the fields

**Figure 89: InfoSets**

From SAP Release 4.6C, *InfoSet* replaces the name valid until then, which is *functional area*.

An InfoSet is a special view of a dataset (logical database, table join, table, sequential file) and is used by SAP Query as a data source. InfoSets, therefore, determine the tables or fields in these tables that can be referenced by a report. In most cases, InfoSets are based on logical databases. SAP Query features a component for maintaining InfoSets. When you create an InfoSet, a data source in an application system is selected. SAP Query is relevant for the Business Information Warehouse because it allows you to define the extract structure by selecting fields in a logical database, table join or other datasets in the form of an InfoSet. In this way, you can use the generic data extraction function for master or transaction data from any InfoSet. When you do so, a query is created for an InfoSet that retrieves data and transfers it to the generic extractor.

InfoSets represent an additional, convenient data source for extracting data generically. Logical databases in all SAP applications, table joins and further datasets can be used as a data source for BI.

**Generic extraction supports:**

- Function modules

Template:

RSAX_BIW_GET_DATA_SIMPLE

Create DataSource for Transactn data: PM_FUNCTION_MODULE

Generic Delta

DataSource	PM_FUNCTION_MODULE	Extraction from View
Applic. Component	T_BW	Extraction from Query
Obj. status	New	Extraction by FM

Texts

Short description	PM_FUNCTION_MODU...
Medium description	PM_FUNCTION_MODULE
Long description	PM_FUNCTION_MODULE

Extraction from DB View

ViewTable	
ExtractStruct.	

Extraction frm SAP Query

InfoSet	
---------	--

Extraction by Function Module

Function Module	
ExtractStruct.	

Figure 90: Generic Extraction with Function Module

A template is provided with which you can extract data with a function module. The data has to be copied from the function module to an interface table E_T_DATA.

Exercise 6: Generic DataSources for Attributes/Texts (Optional)

Exercise Objectives

After completing this exercise, you will be able to:

- Create generic DataSources for attributes and texts
- Load attribute and text data into BI using these DataSources

Business Example

Your company wants to analyze the purchase requisitions created in the SAP OLTP system in BI. Since Business Content does not include a DataSource for this purpose, you have been given the task of generating one for purchasing requisitions using the tools for generic data extraction.

Similarly, reports are also to be supported using the old material number that was transferred from the legacy system to the material master in the SAP OLTP system.

The old material number is to be maintained as a material attribute. In order to do so, you have to generate a new InfoSource and DataSource for the *material* characteristic with the attribute *old material number*.

Task 1:

In order to fill the master data tables (attributes and texts) of a new material characteristic, which will be created in task 2, two new generic DataSources have to be created in the OLTP source system which will extract the material attributes and material texts. These DataSources will be based on two different data base views which first have to be created in the OLTP system.

1. Create a new database view **ZMAATT##** with the short description **Attributes material number GR##** in the ABAP Workbench Dictionary on your SAP OLTP System.
2. Integrate the *client*, *material number*, and *old material number* fields from the table **MARA** as fields in your view. Then activate the view as a local object.
3. Now create a new database view **ZMATXT#** with the short description **Texts material number GR##** in the ABAP Workbench Dictionary for the texts of your new characteristic.
4. The table **MAKT** is to be used as the data source. The *client*, *material number*, *language keys* and *material description* in this table are required as view fields.

Continued on next page

5. Once you have defined the data base views, they can be used as extractors for BI when the new DataSources are created. To create the DataSources, go to the BI IMG (transaction SBIW) from your SAP OLTP System and create the new generic DataSource **ATTRGR##** to load the new BI InfoObject **MAT_GR##** which will be created in Task 2.

Note the following regarding the new DataSource **ATTRGR##**: Later, it should be located in the *DataSources* tree of the BI system, under the Application Component **BW350-##**. The extraction structure is based on the view **ZMAATT##** you defined previously.

Field Name	Enter
<i>Application Comp.</i>	BW350-##
<i>Short Description</i>	OldMat GR##
<i>Medium Description</i>	Old material no. GR##
<i>Long Description</i>	Old material number GR##
<i>View/Table</i>	ZMAATT##

Save your new DataSource, select the fields *MATNR* and *BISMT* as selection fields and go back to the screen you came from.

6. Follow the same procedure to create the DataSource for texts.

The new DataSource **TXTGR##** should be located later in the *DataSources* tab of the BI System, under the Application Component **BW350-##**. The extraction structure is based on the view **ZMATXT##** you defined previously.

Field Name	Enter
<i>Application Comp.</i>	BW350-##
<i>Short Description</i>	GR## - texts
<i>Medium Description</i>	Old material no. GR## - texts
<i>Long Description</i>	Old material number GR## - texts
<i>View/Table</i>	ZMATXT##

Save your new DataSource, select the fields *MATNR* and *SPRAS* as selection fields and go back to the screen you came from.

Continued on next page

Task 2:

Once you have fulfilled all of the necessary prerequisites in the SAP OLTP System, the DataSources must be replicated. After the new characteristic for the material number was created the attribute and text data can be loaded into the BI System.

1. In the Data Warehousing Workbench of your BI System, carry out a Data Source Replication process in the foreground. Please ensure you access the replication path under your Application Component **BW350-##**.



Note: Please note: **DO NOT** replicate via the context menu of the root node of the Source System **T90CLNT090**.

Do not forget to activate the replicated DataSources **ATTRGR##** and **TXTGR##!**

2. Now you will have to create an InfoObject **MAT_GR##** for the material number which will be the data target for the previously created master data DataSources. Create your InfoObject **MAT_GR##** in your InfoObject Catalog **T_BW350C##** created in a previous exercise of this course.

Field Name	Enter
<i>Char</i>	MAT_GR##
<i>Long Description</i>	Material Number GR##
<i>Data type</i>	CHAR
<i>Length</i>	18
<i>Conersv. Rout.</i>	-

Define the InfoObject **OLDMAT##** as an attribute of InfoObject **MAT_GR##**.

Note that the InfoObject **OLDMAT##** does not presently exist in the BI system. When entering InfoObject **OLDMAT##** as an Attribute of InfoObject **MAT_GR##** you will be presented with the “Create Attribute” dialog box. In this dialog box, select *Create Attribute as Characteristic* and in the following *Create Characteristic* screen enter the following information:

Field Name	Enter
<i>Long Description</i>	Old Material Number GR##
<i>Data type</i>	CHAR
<i>Length</i>	18
<i>Conersv. Rout.</i>	-

Continued on next page

3. Create new Transformations between your DataSources **ATTRGR##** and **TXTGR##** respectively and the InfoObject **MAT_GR##** in order to be able to upload attributes and text data into the master data tables of InfoObject **MAT_GR##**.

 **Note:** Note that the InfoObject **MAT_GR##** needs to be designated as an InfoProvider before the Transformations can be created. Therefore you have to assign the InfoObject to your InfoArea **T_BW350_GR##**.
4. Create an InfoPackage for DataSource **ATTRGR##** for use in loading attribute data to the PSA. Use **MAT_GR## Load Attributes** as InfoPackage description and *Start Data Load immediately*.

Check your data load in the Data Load Monitor.
5. Create an InfoPackage for DataSource **TXTGR##** for use in loading attribute data to the PSA. Use **MAT_GR## Load Texts** as InfoPackage description and *Start Data Load immediately*.

Check your data load in the Data Load Monitor.
6. Create a Data Transfer Process between your DataSource **ATTRGR##** and your InfoObject **MAT_GR##**.

Restrict the **Full** update to the interval **M-01** to **M-10** for the field *Material* and execute the Data Transfer Process. Check your data load in the DTP Monitor.
7. Create a Data Transfer Process between your DataSource **TXTGR##** and your InfoObject **MAT_GR##**.

Restrict the **Full** update to the interval **M-01** to **M-10** for the field *Material* and execute the Data Transfer Process. Check your data load in the DTP Monitor.
8. Now take a look at the master data for the material **M-01** using the function *Maintain Master Data* in the InfoObjects tree in Data Warehousing Workbench.

Solution 6: Generic DataSources for Attributes/Texts (Optional)

Task 1:

In order to fill the master data tables (attributes and texts) of a new material characteristic, which will be created in task 2, two new generic DataSources have to be created in the OLTP source system which will extract the material attributes and material texts. These DataSources will be based on two different data base views which first have to be created in the OLTP system.

1. Create a new database view **ZMAATT##** with the short description **Attributes material number GR##** in the ABAP Workbench Dictionary on your SAP OLTP System.
 - a) *Data Warehousing Workbench → Modeling → Source Systems → Context Menu for the Source System 'T90CLNT090': Customizing Extractors → Transaction: /OSE11*

Field Name	Enter
<i>View</i>	✓
<i>Object Name</i>	ZMAATT##

Choose *Create* .

Field Name	Enter
<i>Database view</i>	✓

Choose *Copy* .

Field Name	Enter
<i>Short Description</i>	Attribute material number GR##

Continued on next page

2. Integrate the *client*, *material number*, and *old material number* fields from the table **MARA** as fields in your view. Then activate the view as a local object.
- a) Enter the table **MARA** in the tables column on the left side of the screen

Field Name	Enter
Table	MARA

Select the *FLDS* tab page

Select the *Table Fields* pushbutton

Select the following fields

Field Name	Enter
<i>MANDT</i>	Client
<i>MATNR</i>	Material number
<i>BISMT</i>	Old material number

Choose *Copy*  and then *Activate* .

Choose *Local Object*

Choose *Back* 

Continued on next page

3. Now create a new database view **ZMATXT#** with the short description **Texts material number GR##** in the ABAP Workbench Dictionary for the texts of your new characteristic.

a) SAP OLTP System → Transaction: /OSE11

Field Name	Enter
Object Name	ZMATXT##
View	✓

Choose *Create* 

Field Name	Enter
Database View	✓

Choose *Transfer (Enter)* 

Field Name	Enter
Short Description	Text material number GR##

Continued on next page

4. The table **MAKT** is to be used as the data source. The *client, material number, language keys* and *material description* in this table are required as view fields.

- a) Enter the table **MAKT** in the tables column on the left side of the screen

Field Name	Enter
Table	MAKT

Select the *FLDS* tab page

Select the *Table Fields* pushbutton

Select the following fields

Field Name	Enter
MANDT	Client
MATNR	Material number
SPRAS	Language keys
MAKTX	Material description

Choose *Copy* and then *Activate* .

Choose *Local Object*

Choose *Back*

5. Once you have defined the data base views, they can be used as extractors for BI when the new DataSources are created. To create the DataSources, go to the BI IMG (transaction SBIW) from your SAP OLTP System and create the new generic DataSource **ATTRGR##** to load the new BI InfoObject **MAT_GR##** which will be created in Task 2.

Note the following regarding the new DataSource **ATTRGR##**: Later, it should be located in the *DataSources* tree of the BI system, under the Application Component **BW350-##**. The extraction structure is based on the view **ZMAATT##** you defined previously.

Field Name	Enter
Application Comp.	BW350-##
Short Description	OldMat GR##

Continued on next page

Field Name	Enter
<i>Medium Description</i>	Old material no. GR##
<i>Long Description</i>	Old material number GR##
<i>View/Table</i>	ZMAATT##

Continued on next page

Save your new DataSource, select the fields *MATNR* and *BISMT* as selection fields and go back to the screen you came from.

- a) SAP OLTP System → Transaction: /OSBIW → Generic DataSources
→ Maintain Generic DataSource

Choose *Execute* 

Field Name	Enter
<i>Radio Button: Master Data Attributes</i>	
<i>Master Data Attributes</i>	ATTRGR##



Note: Note the following regarding the new DataSource **ATTRGR##**: Later, it should be located in the DataSource tab of the BI System, under the application component **BW350-##**. The extraction structure is based on the view **ZMAATT##** you defined previously.

Choose *Back* 

- b) Choose *Create*  Within the *Create DataSource for Master Data Attribs* screen specify an Application Component, short, medium and long text as well as the source data base view as given below.

Field Name	Enter
<i>Application Comp.</i>	BW350-##
<i>Short Description</i>	OldMat GR##
<i>Medium Description</i>	Old material no. GR##
<i>Long Description</i>	Old material number GR##
<i>View/Table</i>	ZMAATT##

- c) Choose *Save* 
d) Choose *Local Object*.
e) Select the checkbox *Selection* for fields *MATNR* and *BISMT* as selection fields
f) Choose *Save* 
g) Choose 

6. Follow the same procedure to create the DataSource for texts.

Continued on next page

The new DataSource **TXTGR##** should be located later in the *DataSources* tab of the BI System, under the Application Component **BW350-##**.
 The extraction structure is based on the view **ZMATXT##** you defined previously.

Field Name	Enter
<i>Application Comp.</i>	BW350 -##
<i>Short Description</i>	GR## - texts
<i>Medium Description</i>	Old material no. GR## - texts
<i>Long Description</i>	Old material number GR## - texts
<i>View/Table</i>	ZMATXT##

Save your new DataSource, select the fields *MATNR* and *SPRAS* as selection fields and go back to the screen you came from.

- a) SAP OLTP System → Transaction: /NRSO2:

Field Name	Enter
<i>Radio Button: Texts</i>	✓
<i>Texts</i>	TXTGR##

Choose *Create* 

- b) Choose *Save* 
- c) Choose *Local Object*
- d) Select the checkbox *Selection* for fields *MATNR* and *SPRAS*
- e) Choose *Save* 
- f) Choose *Back* 

Continued on next page

Task 2:

Once you have fulfilled all of the necessary prerequisites in the SAP OLTP System, the DataSources must be replicated. After the new characteristic for the material number was created the attribute and text data can be loaded into the BI System.

1. In the Data Warehousing Workbench of your BI System, carry out a Data Source Replication process in the foreground. Please ensure you access the replication path under your Application Component **BW350-##**.



Note: Please note: **DO NOT** replicate via the context menu of the root node of the Source System **T90CLNT090**.

Do not forget to activate the replicated DataSources **ATTRGR##** and **TXTGR##**!

- a) *Data Warehousing Workbench → Modeling → Source Systems → SAP → Context Menu for T90CLNT090 → Display DataSource Tree and find your Application Component **BW350-##**.*
 - b) From the context menu of your Application Component **BW350-##** choose *Replicate Metadata*.
 - c) In the dialog box *Unknown DataSource*, you will be asked “This DataSource does not yet exist in the BI system. How do you want to create the object in BI?”
Accept the default *As DataSource* and choose continue.
 - d) After having replicated your DataSources, use the context menu of your DataSource **ATTRGR##** and choose *Change*. In the *Change DataSource* screen activate your DataSource by choosing the Activate icon.
Choose the Back icon.
 - e) Then use the context menu of your DataSource **TXTGR##** and choose *Change*. In the *Change DataSource* screen activate your DataSource by choosing Activate icon. Choose the Back icon.
2. Now you will have to create an InfoObject **MAT_GR##** for the material number which will be the data target for the previously created master data DataSources. Create your InfoObject **MAT_GR##** in your InfoObject Catalog **T_BW350C##** created in a previous exercise of this course.

Field Name	Enter
<i>Char</i>	MAT_GR##
<i>Long Description</i>	Material Number GR##

Continued on next page

<i>Data type</i>	CHAR
<i>Length</i>	18
<i>Conersv. Rout.</i>	-

Define the InfoObject **OLDMAT##** as an attribute of InfoObject **MAT_GR##**.

Note that the InfoObject **OLDMAT##** does not presently exist in the BI system. When entering InfoObject **OLDMAT##** as an Attribute of InfoObject **MAT_GR##** you will be presented with the “Create Attribute” dialog box. In this dialog box, select *Create Attribute as Characteristic* and in the following *Create Characteristic* screen enter the following information:

Continued on next page

Field Name	Enter
Long Description	Old Material Number GR##
Data type	CHAR
Length	18
Conersv. Rout.	-

- a) Use the menu *Data Warehousing Workbench → Modeling → InfoObjects → BW Training → BW350 → T_BW350_GR## InfoArea → Context Menu of your InfoObject Catalog T_BW350C##*
- b) From the context menu of your InfoObject Catalog **T_BW350C##**, choose *Create InfoObject* and enter the following information:

Field Name	Enter
Char	MAT_GR##
Long Description	Material Number GR##

- c) Choose Transfer (Enter ).
- d) On the *General* tab page, choose:

Field Name	Enter
Data type	CHAR
Length	18
Conersv. Rout.	-

- e) On the *Attributes* tab page, enter the following attribute:

Field Name	Enter
Attributes	OLDMAT##

and choose Enter .

Continued on next page

3. Create new Transformations between your DataSources **ATTRGR##** and **TXTGR##** respectively and the InfoObject **MAT_GR##** in order to be able to upload attributes and text data into the master data tables of InfoObject **MAT_GR##**.



Note: Note that the InfoObject **MAT_GR##** needs to be designated as an InfoProvider before the Transformations can be created. Therefore you have to assign the InfoObject to your InfoArea **T_BW350_GR##**.

- a) Use the menu *Data Warehousing Workbench → Modeling → DataSources*

Locate your DataSource **ATTRGR##** and from the context menu choose *Create Transformation*.

- b) In the diaglog box, *Create Transformationenter*:

Field Name	Enter
<i>Object Type</i>	InfoObject
<i>Subtype of Object</i>	Attributes
<i>Name</i>	MAT_GR##

and then choose, *Transfer (Enter)*

In the *Assign InfoArea* screen you have to designate your InfoObject **MAT_GR##** as an InfoProvider. Enter your InfoArea **T-BW350_GR##** and choose *Assign InfoArea (Enter)*.

- c) Assign the InfoObject **MAT_GR##** to the *MATNR* field and the InfoObject **OLDMAT##** to the *BISMT* field by clicking on the field on the left, holding the mouse button down and dragging to the appropriate field on the right and releasing the button.

- d) In the *Assign InfoArea* screen you have to designate your InfoObject **MAT_GR##** as InfoProvider. Enter your InfoArea **T_BW350_GR##** and choose *Assign InfoArea (Enter)*.

- e) Click the activate icon .

- f) Use the menu *Data Warehousing Workbench → Modeling → DataSources*

Locate your DataSource **TXTGR##** and from the context menu choose *Create Transformation*.

- g) In the diaglog box, *Create Transformationenter*:

Continued on next page

Field Name	Enter
<i>Object Type</i>	InfoObject
<i>Subtype of Object</i>	Texts
<i>Name</i>	MAT_GR##

and then choose, *Transfer (Enter)* 

- h) Assign the InfoObject **MAT_GR##** to the *MATNR* field, the InfoObject **0LANGU** to the *SPRAS* field and the InfoObject **0TXTSH** to the *MAKTX* field by clicking on the field on the left, holding the mouse button down and dragging to the appropriate field on the right and releasing the button.
 - i) Click the activate icon .
4. Create an InfoPackage for DataSource **ATTRGR##** for use in loading attribute data to the PSA. Use **MAT_GR## Load Attributes** as InfoPackage description and *Start Data Load immediately*.

Check your data load in the Data Load Monitor.

- a) Locate your DataSource **ATTRGR##** and from the context menu choose *Create InfoPackage* and in the presented dialog enter the name **MMAT_GR## Load Attributes**. Then choose Save  icon.
- b) Go to the *Extraction* tab to make sure that the entry in the *Adapter* field is **Access to SAP Data through Service API**.
- c) On the *Processing* tab you will see that the option **Only PSA** is not available for change.
 **Note:** In BI, InfoPackages only extract data to the PSA table.
- d) On the *Update* tab the only option should be a **Full update**.
- e) Ordinarily, the InfoPackage would not be scheduled on the *Schedule* tab, but the InfoPackage would be included in a Process Chain. In order to save time, in this exercise, we will not utilize a Process Chain, so on the *Schedule* tab, make sure the option **Start Data Load Immediately** is selected and then click the button *Start*.
- f) Within InfoPackage Maintenance (Scheduler) choose the Monitor .
- g) On the tab page, choose *Detail* and the pushbutton Refresh Details  until the load process is finished.

Continued on next page

5. Create an InfoPackage for DataSource **TXTGR##** for use in loading attribute data to the PSA. Use **MAT_GR## Load Texts** as InfoPackage description and *Start Data Load immediately*.

Check your data load in the Data Load Monitor.

- a) Locate your DataSource **TXTGR##** and from the context menu choose *Create InfoPackage* and in the presented dialog enter the name **MMAT_GR## Load Texts**. Then choose Save  icon.
- b) Go to the *Extraction* tab to make sure that the entry in the *Adapter* field is **Access to SAP Data through Service API**.
- c) On the *Processing* tab you will see that the option **Only PSA** is not available for change.
 **Note:** In BI, InfoPackages only extract data to the PSA table.
- d) On the *Update* tab the only option should be a **Full update**.
- e) Ordinarily, the InfoPackage would not be scheduled on the *Schedule* tab, but the InfoPackage would be included in a Process Chain. In order to save time, in this exercise, we will not utilize a Process Chain, so on the *Schedule* tab, make sure the option **Start Data Load Immediately** is selected and then click the button *Start*.
- f) Within InfoPackage Maintenance (Scheduler) choose the Monitor  icon.
- g) On the tab page, choose *Detail* and the pushbutton Refresh Details  until the load process is finished.

6. Create a Data Transfer Process between your DataSource **ATTRGR##** and your InfoObject **MAT_GR##**.

Continued on next page

Restrict the **Full** update to the interval **M-01** to **M-10** for the field *Material* and execute the Data Transfer Process. Check your data load in the DTP Monitor.

- a) Use the menu *Data Warehousing Workbench* → *Modeling* → *DataSources*

Locate your DataSource **ATTRGR##** and from the context menu choose *Create Data Transfer Process*.

- b) In the diaglog box, *Creation of Data Transfer Process*enter:

Field Name	Enter
<i>Data Transfer Proc.</i>	MAT_ATTR_GR##
<i>Object Type</i>	InfoObject
<i>Subtype of Object</i>	Attributes
<i>Name</i>	MAT_GR##

and then choose, *Transfer (Enter)* 

- c) On the *Extractiontab*, change the *Extraction Mode* to **Full**.
Click the *Filter* button and restrict the update to the interval **M-01** to **M-10** for the field *Material*.
 - d) On the *Update* tab and on the *Execute* tab, accept the defaults and activate the Data Transfer Process by using the Activate  icon.
 - e) On the *Executetab*, click the *Execute* button to execute the Data Transfer Process.
7. Create a Data Transfer Process between your DataSource **TXTGR##** and your InfoObject **MAT_GR##**.

Continued on next page

Restrict the **Full** update to the interval **M-01** to **M-10** for the field *Material* and execute the Data Transfer Process. Check your data load in the DTP Monitor.

- a) Use the menu *Data Warehousing Workbench* → *Modeling* → *DataSources*

Locate your DataSource **ATTRGR##** and from the context menu choose *Create Data Transfer Process*.

- b) In the diaglog box, *Creation of Data Transfer Processenter*:

Field Name	Enter
<i>Data Transfer Proc.</i>	MAT_TXT_GR##
<i>Object Type</i>	InfoObject
<i>Subtype of Object</i>	Texts
<i>Name</i>	MAT_GR##

and then choose, *Transfer (Enter)*

- c) On the *Extractiontab*, change the *Extraction Mode* to **Full**. Click the *Filter* button and restrict the update to the interval **M-01** to **M-10** for the field *Material*.
- d) On the *Updatetab* and on the *Execute tab*, accept the defaults and activate the Data Transfer Process by using the Activate icon.
- e) On the *Executetab*, click the *Execute* button to execute the Data Transfer Process.

8. Now take a look at the master data for the material **M-01** using the function *Maintain Master Data* in the InfoObjects tree in Data Warehousing Workbench.

- a) Use the menu *Data Warehousing Workbench* → *Modeling* → *InfoObjects* → Use the “Binoculars” symbol to find your InfoObject **MAT_GR##** → Context Menu for InfoObject **MAT_GR##** → *Maintain Master Data*

Field Name	Enter
<i>MATERIAL NUMBER GR##</i>	M-01

Choose the Execute icon.



Lesson Summary

You should now be able to:

- Create generic DataSources
- Use different data acquisition methods to create generic DataSources
- Describe the emulation process of existing generic and non-generic DataSources

Lesson: Enhancement of Business Content DataSources

Lesson Overview

In this lesson, we will demonstrate the enhancement options available for a DataSource. This is valid for both master data and transaction data. Beforehand, though, you must check whether to enhance an existing DataSource or create a new one.



Lesson Objectives

After completing this lesson, you will be able to:

- Enhance Business Content DataSources according to the demands of your company
- Implement the data enhancements using the function enhancements of the service API
- Determine the function enhancements in the individual applications

Business Example

Your company wants to include the old material number as an additional characteristic for certain DataSources from logistics. Some of the standard extractors will have to be modified as a result. You decide to enhance these extractors in Business Content.

Your company also wants the materials assigned to a class of type 001 in the OLTP system to be displayed as such in the material reports carried out in BI. Here too, you decide to enhance Business Content (0MATERIAL).

Enhancement of Business Content DataSources

need introduction here



Additional information in master data and transaction data (characteristics, key figures) is required for reporting.

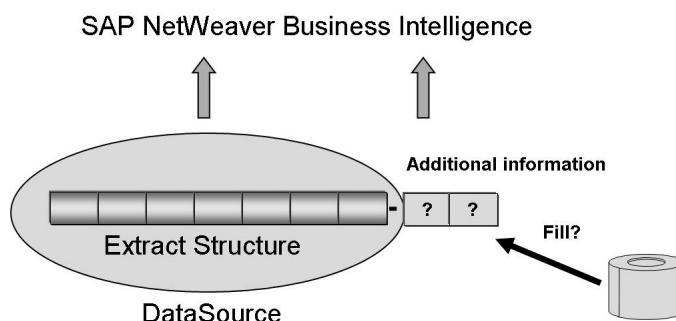


Figure 91: Enhancing DataSources in Business Content

There can be various reasons why the Business Content delivered by SAP does not contain sufficient information. It may be that you are also not working with the SAP standard in the source system and you have to modify the data extraction. Thus it is absolutely necessary that the additional data is available at the time of extraction and that all required keys for reading the additional data are part of the original data.

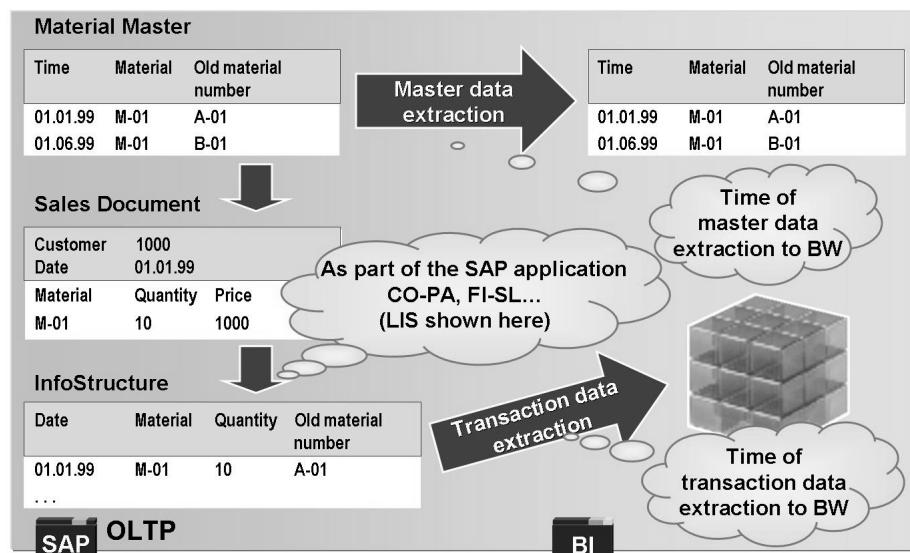


Figure 92: Data Enhancement in the Extraction Path

In both the master data and in extraction of transaction data, it may be necessary to enhance the data to be extracted with new information.

Some application areas or information systems such as the LIS or CO-PA allow you to enhance characteristics that are not contained in the documents by means of enhancements or other methods. These types of characteristics are also determined when a document is posted (for more information on LIS data extraction and CO-PA extraction, see the appendix).

If the data should or must be enhanced at the time of BI data extraction, this has to be done by enhancing the DataSource.



In most cases, enhancements are made through “User Exits” either in the SAP applications or on the way to BW.

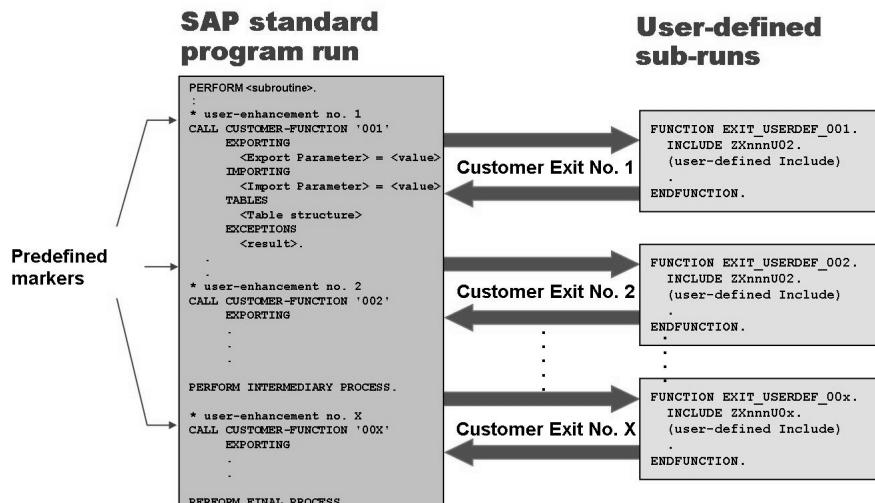


Figure 93: Enhancements using Customer Exits

Customer exits are generally used for the actual enhancement of the data. Customer exits are used to branch to predefined points from the SAP standard program run into user-defined subprograms. This enhances the standard functioning with of the customer's own aspects. This gives SAP software the highest possible flexibility. The user-defined codes are managed as isolated objects in the customer namespace, so there is no worry of complications with the next release or when correction statuses are implemented.

In the example above, the section between the dotted lines described what data the user code can use for its logic and what the SAP code expects to receive back from the customer's logic.

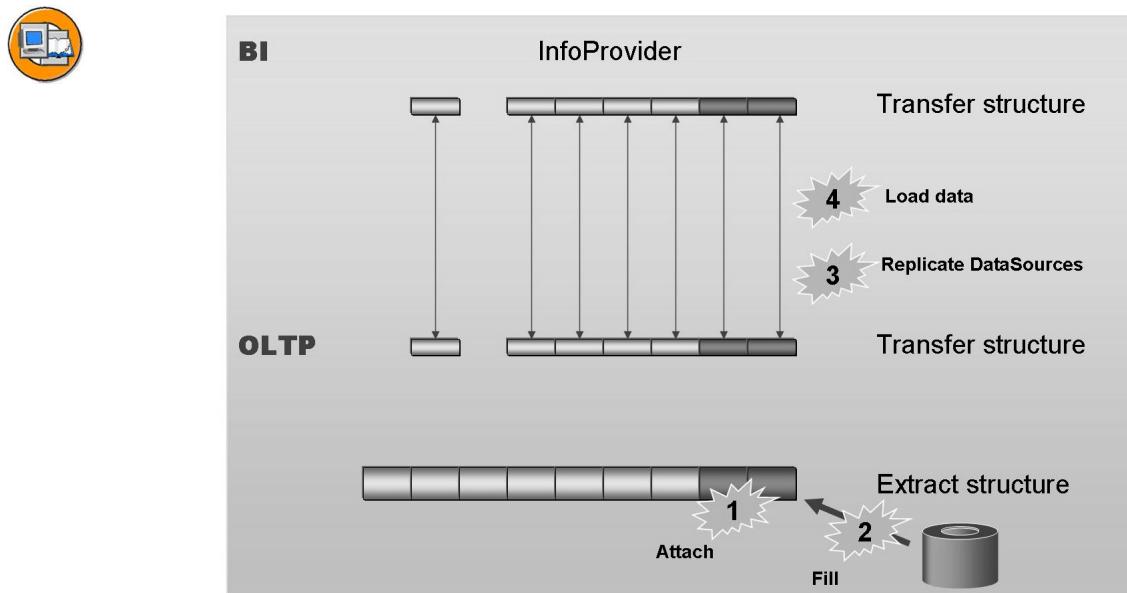


Figure 94: Enhanced Extraction to BI

If your requirements are not covered completely by the DataSources supplied with Business Content, you can obtain additional information in a routine by developing your own program and integrating it in a function enhancement.

To integrate a program in a function enhancement, follow these steps:

- Define the required fields in an append structure that is appended to the extract structure of your DataSource.
- Write your function exit to call up the relevant data sources for your DataSource.
- Replicate the DataSource in BI.
- Extract the data for your enhanced DataSource.

→ **Note:** Beforehand, check whether it might be less work to create a new generic DataSource instead of enhancing an existing one.

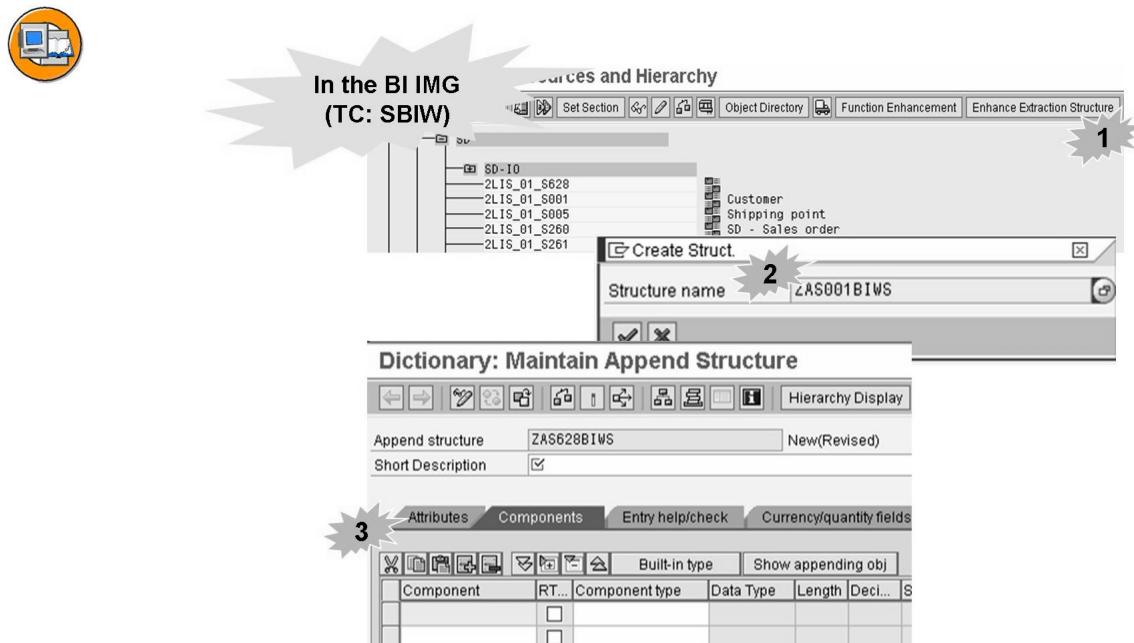


Figure 95: Creating and Maintaining Customer Appends

To enhance the extraction structure, in the IMG in the source system (transaction SBIW), choose *Business Information Warehouse → Postprocessing of DataSources → Edit DataSources and Application Component Hierarchy*.

First, you have to generate the customer append for the extract structure of your DataSources. When you do so, the system proposes a name starting with **Z**A followed by the name of the extraction structure. Maintain the fields that you want to enhance later. All of these fields should start with **ZZ** as a way of identifying their special purpose. Finally, generate your append structure.



Caution: The extraction structure from the LO Cockpit cannot be enhanced. An enhancement using transaction SE11 is not specific to BI and cannot be used for BI. In this case it would be worthwhile to enhance extraction during the Data Transfer Process with a routine to fill the field.

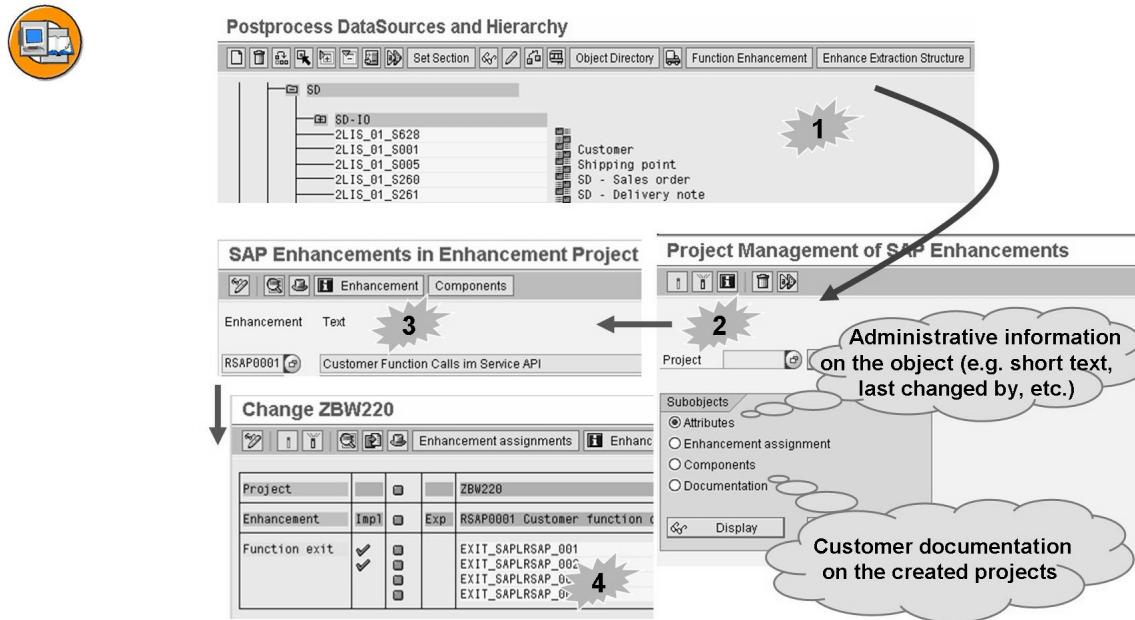


Figure 96: Developing Function Enhancements during BI Extraction

In the second step, you must define the function enhancement for filling the new fields in the extraction structure.

You have to create a project before you can use an enhancement like this. You can then integrate one or more enhancements into this project. For our task, SAP provides the enhancement RSAP0001. Remember that an enhancement cannot be used in two projects simultaneously.

There are four different enhancement components that can be used for the enhancement RSAP0001. Their Includes contain the code for filling the extraction structure.

- EXIT_SAPLRSAP_001 Transaction data supply
- EXIT_SAPLRSAP_002 Master data and text supply
- EXIT_SAPLRSAP_003 Texts supply
- EXIT_SAPLRSAP_004 Hierarchy supply

Every enhancement project has documentation, which also explains the individual parameters of the function module. In order to use the enhancement, it must be activated. To display documentation on the entire enhancement, choose *Utilities* → *Display Entire Docu.* in the Project Management of SAP Enhancements transaction CMOD.



Project Management of SAP Enhancements

```

tables:    mara.
data:    l_s001biws like s001biws,
         l_tabix like sy-tabix.

case i_source.
when '2LIS_01_S001'.

* Find material in the master data table MARA
loop at c_t_data into l_s001biws.
l_tabix = sy-tabix.
select single * from mara where matnr = l_s001biws-matnr.

* Fill fields with data from the master data table MARA
if sy-subrc = 0.
l_s001biws-zzbismt = mara-bismt.
modify c_t_data from l_s001biws index l_tabix.
endif.
endloop.
endcase.

```

Figure 97: Sample Code

The graphic above shows an example of code for a function enhancement.



Hint: Always use a CASE statement to query the DataSource because this function exit is called with every DataSource.

This example illustrates only the procedure for determining a field value. We did not focus on performance-optimized programming. If the data volume to be expected is very high, buffers can be used for the material number, block reads, DB buffer mechanisms, field symbols, and so on can be used to optimize the runtime.

Once the function exit is used, it is called with each extraction (CASE statement required). The following prerequisites and restrictions are valid:

- Key fields for the enhanced Content must be available.
- Customer exit is only specific to the append fields.
- The maximum field length is 60 characters.



Note: Do not change the field contents and do not add data records.

Be sure that all recommended units of measure are included.

Further documentation is provided under the transaction CMOD.

Troubleshooting in enhancements

Extractors are executed as background tasks. This means that breakpoints do not have any effect.

Prerequisites for troubleshooting the source system

- Development authorization in the source system
- Troubleshooting authorization in the source system

Follow these steps to troubleshoot the source system:

1. Log on to the source system.
2. Test and debug in extractor checks (RSA3).
3. Monitor running tasks (SM50).
4. Start extraction with the Scheduler in BI.
5. Locate the process in the monitor and start troubleshooting.
6. Restart the Data Warehousing Workbench after code changes in the customer exit.

Exercise 7: Enhancing a DataSource for Transaction Data

Exercise Objectives

After completing this exercise, you will be able to:

- Add to an existing DataSource by extending the extraction structure and filling the field “BISMT (old material number)”.
- Use the “function enhancement” to fill this additional field during the extraction process.

Business Example

Your company operates in the area of LO extraction. You now need reports for the field *BISMT* for the old material number in the SAP OLTP system. At present, this field does not form part of the original extraction structure.

Task 1:

Enhance the DataSource for transaction data **2LIS_02_S9##** (where ## = **30** + pc number [the pc number is either your group number assigned by your Instructor or the number on your workstation]) in your SAP OLTP system.



Hint: The DataSources used for training purposes are based on LIS information structures. The extraction behavior nevertheless corresponds with standard Business Content DataSources.

1. Call the transaction *SBIW* in your SAP OLTP system. Create a customer append with the description **Append Structure for S9##BIWS** and the proposed technical name for your DataSource **2LIS_02_S9##**. Insert the field *ZZBISMT* with the data element *BISMT*. Activate your customer append as a local object.

Confirm the proposed structure **ZAS9##BIWS** (the naming convention in the SAP system requires the prefix **Z** for customer Data Dictionary objects, **A** stands for “append”, and **S9##BIWS** for the extraction structure for your DataSource).

Field Name	Enter
<i>Short Description</i>	Append structure for S9##BIWS
<i>Component</i>	ZZBISMT
<i>Component type</i>	BISMT (old material number)

Continued on next page

2. Now take a look at the coding for the enhancement component “EXIT_SAPLRSAP_001” in the project ZBW220 “Enhancements for BW”.

Why was the code for determining the “old material number” included in a CASE statement?



Hint: Check the code and make sure it refers to DataSources in the CASE statement. Ensure that the code in the CASE statement for your Data Source will be executed by searching for the name in the code. This project should be active at this point, **as it was activated by the instructor.**

Task 2:

After you have satisfied all the prerequisites in the SAP OLTP system, the next step is to replicate the DataSource in the BI system and upload data via your enhanced DataSource.

1. In the Data Warehousing Workbench of your BI system, replicate your DataSource **2LIS_02_S9##** in the foreground for the source system T90CLNT090.



Hint: If you use the menu path below to replicate the DataSource, make sure that you carry out the replication only at the level of your existing DataSource. In this way, only the changes (enhancements) you have made above, and not every DataSource, will be updated.

2. Create an InfoPackage for your DataSource **2LIS_02_S9##** and start a FULL-Upload immediately on the Schedule tab. Use **Load Data including Old Material Number GR##** as InfoPackage description. Data load shall be restricted to Material Number **M-01**.
3. Check your data load in the Data Load Monitor
4. After successful data load check the uploaded data in PSA table. The enhanced field (ZZBISMT) containing the old material number should be filled for material number M-01.

Solution 7: Enhancing a DataSource for Transaction Data

Task 1:

Enhance the DataSource for transaction data **2LIS_02_S9##** (where ## = 30 + pc number [the pc number is either your group number assigned by your Instructor or the number on your workstation]) in your SAP OLTP system.



Hint: The DataSources used for training purposes are based on LIS information structures. The extraction behavior nevertheless corresponds with standard Business Content DataSources.

1. Call the transaction **SBIW** in your SAP OLTP system. Create a customer append with the description **Append Structure for S9##BIWS** and the proposed technical name for your DataSource **2LIS_02_S9##**. Insert the field ZZBISMT with the data element BISMT. Activate your customer append as a local object.

Confirm the proposed structure **ZAS9##BIWS** (the naming convention in the SAP system requires the prefix **Z** for customer Data Dictionary objects, **A** stands for “append”, and **S9##BIWS** for the extraction structure for your DataSource).

Field Name	Enter
<i>Short Description</i>	Append structure for S9##BIWS
<i>Component</i>	ZZBISMT
<i>Component type</i>	BISMT (old material number)

- a) Call the Data Warehousing Workbench and choose the *Source Systems* tab in the Modeling area:

Data Warehousing Workbench → Source Systems → T90CLNT090 → Context menu: Customizing Extractors

In the OLTP system,*Data Transfer to the SAP Business Information Warehouse → Postprocessing of DataSources→ Edit DataSources and application component hierarchy*

Choose *Execute*

Select your DataSource **2LIS_02_S9##** below the Application Component Materials Management (SAP R/3 → MM) and choose the “Enhance Extraction Structure” pushbutton.

Continued on next page

Confirm the proposed structure **ZAS9##BIWS**.

Choose *Transfer (Enter)*

Maintain the append structure **ZAS9##BIWS** according to the information given below.

Field Name	Enter
<i>Short Description</i>	Append structure for S9##BIWS
<i>Component</i>	ZZBISMT
<i>Component type</i>	BISMT (old material number)

Choose *Activate*

Choose *Local Object*

Go back to the *Subsequent Processing for DataSources* screen

Choose *Change DataSource*

Deselect both the checkbox for *Hide Field* for the field ZZBISMT

Choose *Save*

and then *Back*

2. Now take a look at the coding for the enhancement component "EXIT_SAPLRSAP_001" in the project ZBW220 "Enhancements for BW".

Continued on next page

Why was the code for determining the “old material number” included in a CASE statement?



Hint: Check the code and make sure it refers to DataSources in the CASE statement. Ensure that the code in the CASE statement for your Data Source will be executed by searching for the name in the code. This project should be active at this point, **as it was activated by the instructor.**

- a) Call the transaction /OCMOD (= Project Management of SAP Enhancements) in your SAP OLTP system:

Field Name	Enter
Project	ZBW220
Components	✓

Choose the *Display* pushbutton

→ Double-click on the function module exit EXIT_SAPLRSAP_001

→ Double-click on the include ZXRSAU01

Why was the code for determining the “old material number” included in a CASE statement?

Answer:

Since this function module exit is called each time data is extracted, the CASE statement specifies the DataSources for which a data enhancement is to be carried out and how this is carried out for the individual DataSources.

Continued on next page

Task 2:

After you have satisfied all the prerequisites in the SAP OLTP system, the next step is to replicate the DataSource in the BI system and upload data via your enhanced DataSource.

1. In the Data Warehousing Workbench of your BI system, replicate your DataSource **2LIS_02_S9##** in the foreground for the source system T90CLNT090.

 **Hint:** If you use the menu path below to replicate the DataSource, make sure that you carry out the replication only at the level of your existing DataSource. In this way, only the changes (enhancements) you have made above, and not every DataSource, will be updated.

 - a) *Data Warehousing Workbench → Modeling → Source Systems → Context Menu for Source System T90CLNT090: DataSource Tree Display → SAP Application Components → Materials Management → Context Menu for DataSource 2LIS_02_S9## → Replicate Metadata*
2. Create an InfoPackage for your DataSource **2LIS_02_S9##** and start a FULL-Upload immediately on the Schedule tab. Use **Load Data including Old Material Number GR##** as InfoPackage description. Data load shall be restricted to Material Number **M-01**.
 - a) Use the menu *Data Warehousing Workbench → Modeling → Source Systems → Context Menu for Source System T90CLNT090: DataSource Tree Display → SAP Application Components → Materials Management → Context Menu for DataSource 2LIS_02_S9## → Create InfoPackage....*
 - b) Enter the above mentioned InfoPackage Description **Load Data including Old Material Number GR##** and choose Save (Enter) .
 - c) On the *Data Selection* tab page, choose:

Field Name	Enter
<i>MATERIALNUMBER</i>	M-01

- d) On the *Schedule* tab page, choose *Start*.
3. Check your data load in the Data Load Monitor
 - a) Within InfoPackage Maintenance (Scheduler) choose the Monitor .

On the tab page, choose *Detail* and the pushbutton *Refresh Details*  until the load process is finished.

Continued on next page

4. After successful data load check the uploaded data in PSA table. The enhanced field (ZZBISMT) containing the old material number should be filled for material number M-01.
 - a) Within the Data Load Monitor choose the PSA maintenance icon. Then choose Continue (Enter)  to display the uploaded data records of the PSA table.
 - b) Have a look at the column *ZZBISMT*. For material number **M-01** this column should contain the old material number information.



Lesson Summary

You should now be able to:

- Enhance Business Content DataSources according to the demands of your company
- Implement the data enhancements using the function enhancements of the service API
- Determine the function enhancements in the individual applications



Unit Summary

You should now be able to:

- Describe the basic principles for connecting SAP source systems to a BI system
- Describe the role of the BI Service API in data acquisition, extraction and staging
- Describe the transfer process for Business Content DataSources
- Describe how the system decides which DataSource type has to be generated
- Describe the necessary steps to extract data from the Logistics application
- Explain the functions in the Logistics Extraction Structure Customizing Cockpit
- Explain the role played by the restructuring (setup) tables and the delta queue in Logistics data extraction
- Describe the update methods to fill the delta queue
- Create generic DataSources
- Use different data acquisition methods to create generic DataSources
- Describe the emulation process of existing generic and non-generic DataSources
- Enhance Business Content DataSources according to the demands of your company
- Implement the data enhancements using the function enhancements of the service API
- Determine the function enhancements in the individual applications

Unit 4

Delta Management

Unit Overview

Delta management is essential for providing data to a data warehouse in order to enable timely staging of changed data in the source system in the Data Warehouse and to reduce the data transfer into the Data Warehouse to the most relevant data. In addition, delta management will ease access to changed data for the respective application and, in some cases, will make it available for the first time.

This unit explains the most important techniques and problems that arise during management of changed, new or deleted (delta) data records with regard to extraction to BI. This knowledge is required:

- To make required and optional transformations and data saves when modeling a data flow. This will ensure that the data flow is designed to be both as complete as possible and as simple as possible.
- To use the correct data target in the right place when modeling a data flow.
- To better understand the correlation in the administration of BI, ensuring effective, inexpensive operation of your BI solution.
- To understand data content during delta transfer to be better able to evaluate the consistency of the data.



Unit Objectives

After completing this unit, you will be able to:

- Describe important elements of delta management for data extraction to BI
- Explain the role of the update mode and the properties of delta processes
- Make decisions on modeling a data flow dependent on the delta properties of the DataSource to be connected
- Manage the delta process effectively
- Explain the basic concepts of delta management and describe how it works
- Provide examples of how delta management works
- Explain additional aspects of delta management and how they work

Unit Contents

Lesson: Delta Management: Overview	239
Lesson: Update Mode and Delta Process.....	245
Exercise 8: Delta Management: Delta Process	267
Lesson: Examples from Applications and Source Systems.....	277
Exercise 9: Generic Delta for Transaction Data.....	285
Lesson: Additional Features of Delta Data Acquisition	298

Lesson: Delta Management: Overview

Lesson Overview

Delta management is essential for supplying data to a data warehouse. This ensures that data that was changed in the source system is staged in the data warehouse without delay and that only the relevant data is transferred to the data warehouse. In addition, certain functions make it easier to access changed data for the respective application. In some instances, this data would otherwise not even be available.

In the following unit, the most important techniques and problems that arise during management of changed, new or deleted (“delta”) data records will be explained in relation to data extraction to BI. This knowledge is required to

- make both necessary and non-essential transformations and data saves when modeling a data flow. This will ensure that the data flow is designed to be both as complete and as simple as possible
- use the correct data target in the right place when modeling a data flow
- understand correlations in the administration of BI better, ensuring effective, inexpensive operation of your BI solution
- understand data content during the delta transfer to improve your assessment of the consistency of the data



Lesson Objectives

After completing this lesson, you will be able to:

- Describe important elements of delta management for data extraction to BI
- Explain the role of the update mode and the properties of delta processes
- Make decisions on modeling a data flow dependent on the delta properties of the DataSource to be connected
- Manage the delta process effectively

Business Example

Your company has recognized how useful it can be to integrate people, information and business processes in a heterogeneous system landscape and would like to benefit from this.

Practice has shown, though, that loading very large datasets makes considerable demands on hardware and system performance. It is therefore necessary to examine if and how the data records can be loaded into BI with a delta process. It is important to understand the modes of operation and the different variants of a delta loading process.

Delta Management: Overview

The following figure explains which BI source system types support delta management. **Delta management** implies the ability to extract only new or changed data records to the BI system in a separate data request. Technically, this means requesting a “delta update” for a DataSource (see the detailed explanations under “update mode”). The table marked “DQ” = delta queue indicates the source system used by the delta queue to temporarily store new or changed data records, (explained in more detail below).

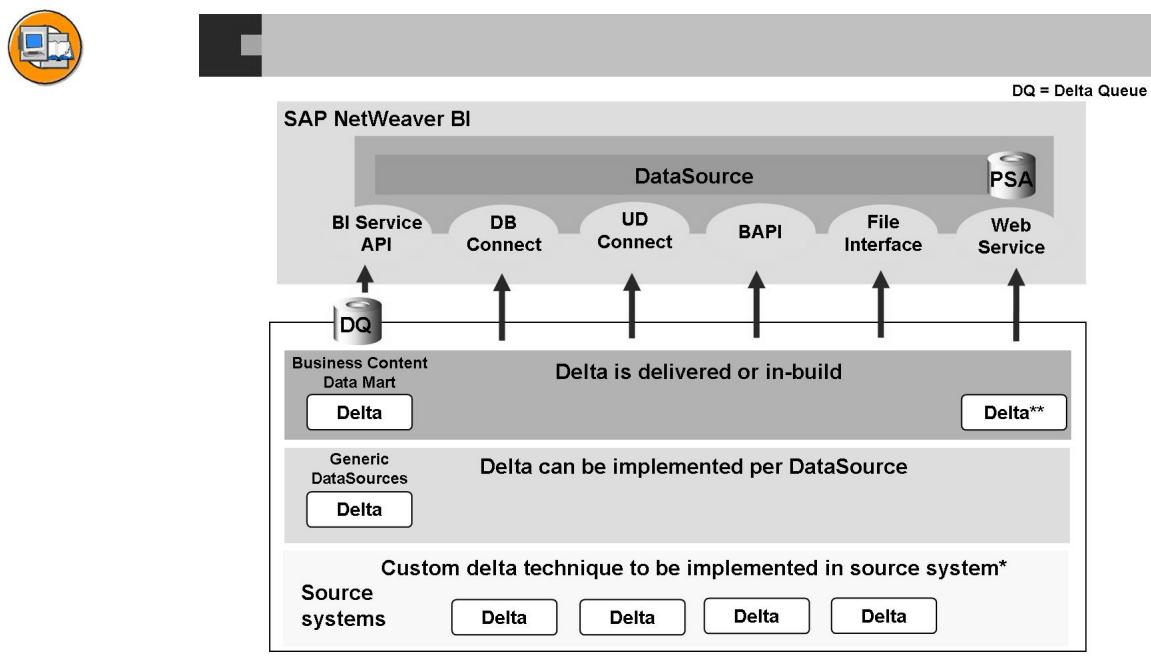


Figure 98: Technical Delta Capabilities of Source Systems in BI

In some instances, however, it is possible to transfer only new or changes data records to the BI system with DataSources that are not capable of providing delta figures. To do this, you have to specify certain selection conditions in the data request. This is not a delta request as such, and is therefore dealt with in the concluding unit “Related Topics”.

The following provides additional notes on several source systems (details of which can be found in the respective units of this course):

- BI source system: The delta queue is not used to transfer data between BI systems if data mart scenarios are being used. If you want to implement generic DataSources capable of providing delta figures on the BI system, these use the delta queue for the corresponding BI (source) system.
- SAP system: Not all Business Content DataSources are capable of providing delta figures. Generic DataSources you have created yourself are not automatically capable of providing delta figures, but can include a delta process with the help of the maintenance transaction for generic DataSources. Certain preconditions are necessary in the underlying data source, however (see the corresponding lesson in the unit “SAP Source Systems”)
- External systems (third-party ETL tools connected to BI by means of BAPIs): The BAPI interface for connecting third-party tools for data acquisition enables the “delta” update mode to be passed on to a third-party data extraction tool. To what extent this delta request can be adapted for or implemented in the partner extraction tool depends on the respective third-party tool, however. Further details lie beyond the scope of this unit.

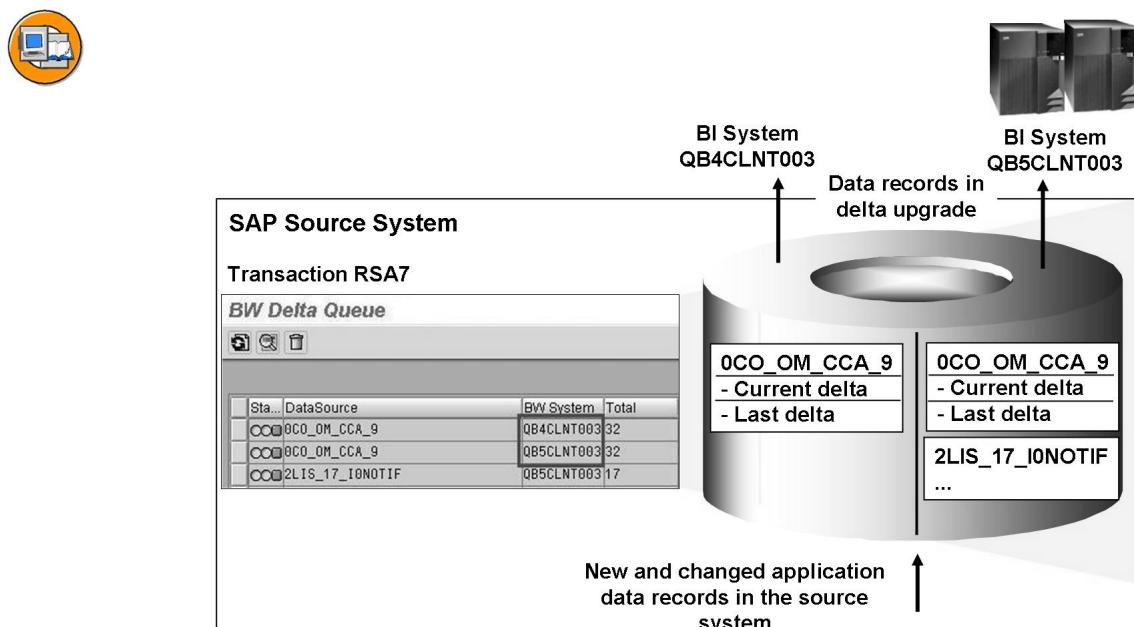


Figure 99: Delta Queue in SAP Source Systems

The delta queue is an S-API (Service API) function. This is the central interface technology used to extract data from SAP source systems to an BI system. Consequently, the delta queue is only used in SAP or BI source systems.

The delta queue is a data store for new or changed data records for a DataSource (that have occurred since the last data request). The new or changed data records are either written to the delta queue automatically using an update process in the source system, or by means of the DataSource extractor when a data request is received from the BI system (details will be provided later).

Central tasks of the data queue

- Storage area for new and changed data (that has appeared since the last delta request) in the source system and will be transferred to the BI system with the next delta request.
- Repeating the previous delta request:
 - The delta queue always contains the current data to be transferred to BI with the next delta request, as well as data from previous delta requests. This is necessary so that delta data records can be re-transferred from the source system if an error occurs during the transfer.
- Independent delta supply for several BI target systems:
 - A separate delta queue for a DataSource in an SAP source system (= logical system name = client in an SAP source system) is held for each target BI system. This enables delta data records for this DataSource to be requested by different BI target systems independently of one another.

The delta queue with transaction data and master data

- Master data DataSources behave in the same way as transaction data DataSources with regard to their delta processes. However, there are individual master data DataSources in current Business Content versions that do not stage their delta data records in the delta queue. In general, though, this behavior refers to master data DataSources with small volumes of data, in which the delta function plays a subordinate role. “Large” master data DataSources for multi-purpose objects such as material, business partner, accounts, and so on, support the delta queue function to the greatest possible extent.
- In addition, some master data DataSources determine their delta using ALE change pointers. This Web AS function allows changes in relation to certain table fields that have to be defined to be “logged jointly”, and works independently of the BI delta queue.

Technical details

In the SAP source system, RSA7 is the central transaction for displaying the delta queue for a DataSource.

The following functions are available here:

- Display the data records in the delta queue for a DataSource. You can select individual data records and choose whether only the current (delta update) data records or those from the previous delta request (delta repetition) are displayed.
- Display the number of LUWs (logical unit of work) that are currently held in the delta queue:

Data records are often written in blocks (= LUW) to the delta queue (in technical terms, a COMMIT). The number next to the delta queue for a DataSource indicates the number of write activities / blocks in the delta queue and not the number of data records contained there! You can determine how the data records are distributed in data packages in a BI (delta) request in Customizing (transaction SBIW in the source system). This has nothing to do with the LUWs.

Delta queues for generic DataSources:

- Display the current status for the delta-relevant field in the generic DataSource. You can delete the delta queue for a DataSource. This removes the delta process for the DataSource irrevocably, since you not only delete any delta data records that may exist, but also the initialization selection for the DataSource. You should therefore take great care when using this function!



Caution: The following technical details are relevant for application and basis administrators, but should only play a minor role when implementing a delta process for a DataSource.

The BI delta queue is based on queue functions in the RFC technology for SAP Web Application Servers (qRFC)

- From a technical perspective, the data is stored in the following tables.
 - TRFCQOUT: Client dependent (sender) pointer table per queue name (DataSource) and destination (recipient) → counter and reference to LUWs in ARFCSDATA
 - ARFCSSSTATE: Link between TRFCQOUT and ARFCSDATA
 - ARFCSDATA: Compressed data for tRFC/qRFC → actual data is stored in clusters
- Alternatively, you can also use the transaction SMQ1 (qRFC monitor (output source)) to display and manage the delta queue.
 - Naming convention for the delta queue for a DataSource:
BI<Client><DataSource>



Lesson Summary

You should now be able to:

- Describe important elements of delta management for data extraction to BI
- Explain the role of the update mode and the properties of delta processes
- Make decisions on modeling a data flow dependent on the delta properties of the DataSource to be connected
- Manage the delta process effectively

Lesson: Update Mode and Delta Process

Lesson Overview

This lesson explains the basic concepts of delta management.



Lesson Objectives

After completing this lesson, you will be able to:

- Explain the basic concepts of delta management and describe how it works

Business Example

Loading very large datasets from a heterogeneous system landscape makes considerable demands on hardware and system performance. It is therefore necessary to examine if and how the data records can be loaded into BI with a delta process.

You need to understand the modes of operation and the different variants of a delta loading process. You also need to understand what is meant by special terms such as delta process.

Update Mode and Delta Process

If you want to implement a delta process, three fundamental questions need to be asked:

1. Does the DataSource to be used have a delta process?
2. In which update mode is data requested?
3. What are the properties of a delta process?



2. In which update mode is data requested?
1. Does the DataSource you are using have a delta process?

3. What are the properties of a delta process?

Figure 100: Update Mode and Delta Process

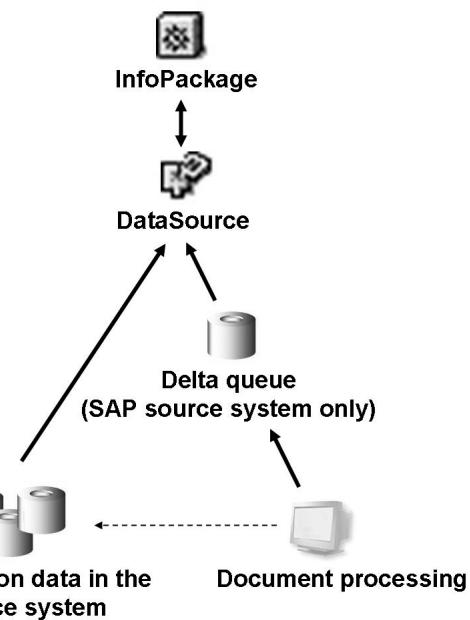
The following graphics describe how you examine a DataSource to ascertain whether or not it can provide delta figures, and how you determine the delta process.



Use the following tools to check whether a DataSource supports a delta process:

- **DataSource postprocessing (in SAP source systems): Transaction RSA6 and SBIW**

- **InfoPackage maintenance in the BI system: If no delta process is available, only "full update" mode is offered in the InfoPackage for the DataSource**



DataSource: Customer version Display

Header Data		Package																																	
DataSource	2LIS_11_VAITH	Description	Sales Document Item Data																																
Extraction	MC11VA01TM																																		
ExtractStruct.	<input checked="" type="checkbox"/>																																		
DirectAccess	<input type="checkbox"/>																																		
Delta Update	<input checked="" type="checkbox"/>																																		
<table border="1"> <thead> <tr> <th>Field Name</th> <th>Short text</th> <th>Selection</th> <th>Hide field</th> <th>Inversion</th> <th>Field only</th> </tr> </thead> <tbody> <tr> <td>ABDIS</td> <td>MRP for delivery schedule types</td> <td><input type="checkbox"/></td> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td><input checked="" type="checkbox"/></td> </tr> <tr> <td>ABGRU</td> <td>Reason for rejection of quotations and sa</td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> </tr> <tr> <td>ABSTA</td> <td>Rejection status for SD item</td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> </tr> <tr> <td>AEDAT</td> <td>Date of Last Change</td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> </tr> </tbody> </table>						Field Name	Short text	Selection	Hide field	Inversion	Field only	ABDIS	MRP for delivery schedule types	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	ABGRU	Reason for rejection of quotations and sa	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	ABSTA	Rejection status for SD item	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AEDAT	Date of Last Change	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Field Name	Short text	Selection	Hide field	Inversion	Field only																														
ABDIS	MRP for delivery schedule types	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>																														
ABGRU	Reason for rejection of quotations and sa	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>																														
ABSTA	Rejection status for SD item	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>																														
AEDAT	Date of Last Change	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>																														

The DataSource supports a delta process

Data Selection Processing Data Targets Update Schedule

Update Mode

Full update

Data Update Type in the Data Targets

Always update data, even if no master data exists

Do not update data if no master data exists for a characteristic

The DataSource does not support a delta process

Figure 101: DataSource: Delta Process 1

The delta process is a property of the extractor and tells you how data will be transferred. As an attribute of the DataSource, it indicates how the data in the DataSource will be transferred to the data target. This enables you to derive the data targets for which a DataSource is suitable, how to update them, and how the serialization has been done, for example.

The type of delta process influences how data is updated to a data target. Serialization is necessary for updating to a DataStore object, as overwriting is supported too. Depending on the delta process, the system decides whether serialization is necessary per request or per data package.

There are various delta processes for SAP source systems (R/3, B2B, CRM, and so on):

1. Creation of deltas with after, before and reverse images that are updated directly to the delta queue.

An **after image** provides the status after the change, a **before image** the status before the change with a minus sign. A **reverse image** sends the record with a minus sign too, indicating at the same time that it is to be deleted. The data packages are serialized in the process. The delta process controls whether adding or overwriting is allowed. In this case, both adding and overwriting is allowed. This process permits data to be updated in a DataStore object and in an InfoCube. (Technical name of the delta process in the system is ABR.)

2. The extractor provides additive deltas that are serialized on a request basis. This serialization is necessary since the extractor provides each key once in a request, and changes to the non-key fields would otherwise not be transferred correctly. It only allows fields to be added. It permits data to be updated in a DataStore object as well as an InfoCube. This delta process is used by LIS DataSources. (Technical name of the delta process in the system: ADD)
3. Creation of deltas with after image that are updated directly to the delta queue.

The data is serialized by package, because the same key can be transferred a number of times within a request. It does not allow data to be updated directly to an InfoCube. To update an InfoCube, you always have to have a DataStore object as a go-between. With numeric key figures, for example, this process only allows you to overwrite data. Adding is not permitted, as this would produce incorrect results. It is used to transfer line items in FI-AP/AR, whilst the variation on this process, in which the extractor can also send records marked for deletion, is used in this function in the BBP. (Technical name of the delta process in the system is AIM or AIMD.)



Check which delta process is supported by the DataSource:

- When questions arise concerning the conception or management of a data flow, it is important to know the properties of a particular delta process.
- Example: Use of DataStore objects (details to follow)**
- Determine the content of the "DELTA" field in tables **RSOLTPSOURCE** or **RSDS** (in BW) or **ROOSOURCE** (in the SAP source system) for the DataSource.

Data Browser: Table ROOSOURCE						
Check Table...						
Table: ROOSOURCE Displayed fields: 18 of 53 Fixed columns: 2 List width 0250						
OBJLTSOURCE	OBJVERS	TYPE	APPLNM	DELTA	STOCKUPD	UPDFL61
OFI_GL_4	A	TRAN	FI-GL	AIE		
2LIS_11_VAITM	A	TRAN	SD	ABR		

Figure 102: DataSource: Delta Process 2

When DataSources are replicated to a BI system, the metadata of the DataSource is stored in one of two tables: either **RSOLTPSOURCE** for DataSources 3.x or **RSDS** for DataSources.

In the table **ROOSOURCE** you can check how the data is extracted using the DataSource. The above view shows only a small number of the fields in the table.

In the “delta” column you will find the delta process that is used. If the field is filled, it means that the DataSource has a delta process. The properties of the delta processes found there are explained later in this unit.

In the “extraction method” column you will find the method by which the data is extracted from the source system:

F1	Function module with its own packaging
V	using view
Q	InfoSet (query)

Furthermore, you can find out which extractors and extraction structures are used to extract the data.

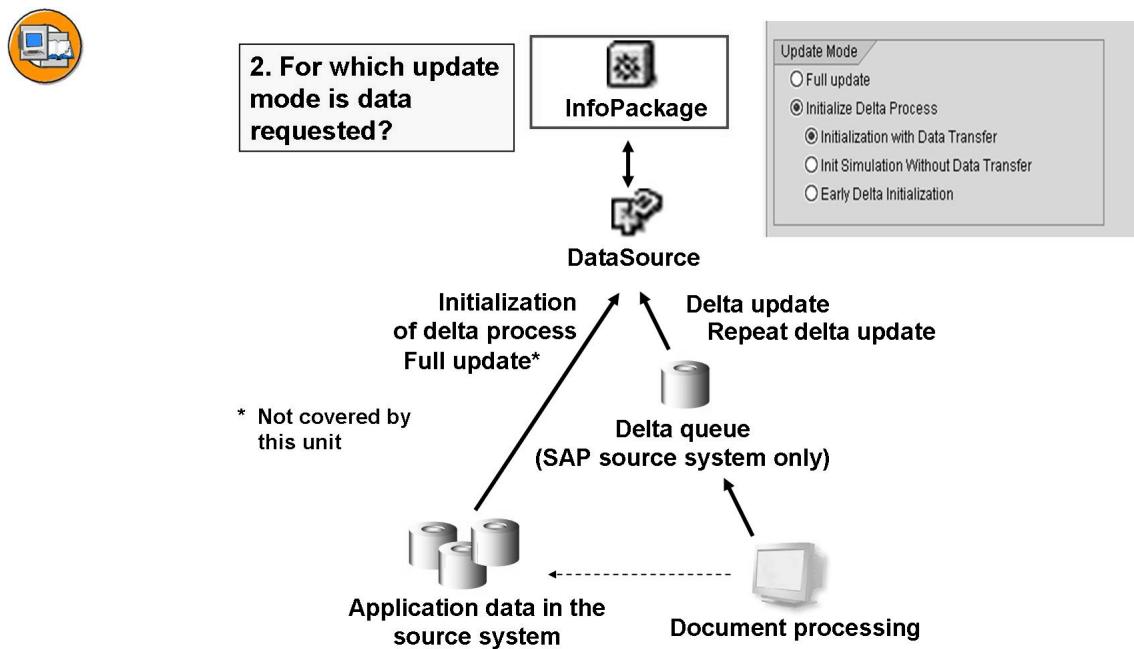


Figure 103: InfoPackage: Update Mode

The update mode in the InfoPackage determines the data you want to request:

- **Full update**

A full update requests all the data that corresponds with the selection criteria you have determined in the scheduler.

You can indicate that a request with full update mode is a full repair request using the scheduler menu. This request can be posted to every data target, even if the data target already contains data from an initialization run or delta for this DataSource/source system combination, and has overlapping selection conditions.

If you use the full repair request to reload the data into the DataStore object after you have selectively deleted data from the DataStore object, for example, note that this can lead to inconsistencies in the data target. If the selections for the repair request do not correspond with the selections you made when selectively deleting data from the DataStore object, posting this request can lead to duplicated data records in the data target, for example.

- **Initialization of the delta process**

Initializing the delta process is a precondition for requesting the delta.

More than one initialization selection is possible for different, non-overlapping selection conditions to initialize the delta process when scheduling InfoPackages for data from an SAP system. This gives you the

option of loading the data relevant for the delta process to the Business Information Warehouse in steps. For example, you could load the data for cost center 1000 to BI in one step and the data for cost center 2000 in another.

The delta requested after several initializations contains the upper amount of all the successful init selections as the selection criterion. After this, the selection condition for the delta can no longer be changed. In the example, data for cost centers 1000 and 2000 were loaded to BI during a delta.

- **Delta update**

A delta update requests only the data that has appeared since the last delta. Before you can request a delta update you first have to initialize the delta process.

A delta update is only possible for loading from SAP source systems.

If a delta update fails (status red in the monitor), or the overall status of the delta request was set to red manually, the next data request is carried out in repeat mode. With a repeat request, the data that was loaded incorrectly or incompletely in the failed delta request is extracted along with the data that has accrued from this point on. A repeat can only be requested in the dialog screen. If the data from the failed delta request has already been updated to data targets, delete the data from the data targets in question. If you do not delete the data from the data targets, this could lead to duplicated data records after the repeat request.

- **Repeat delta update**

If the loading process fails, the delta for the DataSource is requested again.

- **Early delta initialization**

With early delta initialization you can already write the data to the delta queue or to the delta tables in the application while the initialization is requested in the source system. This enables you to initialize the delta process, in other words, the init request, without having to stop posting the data in the source system. You can only carry out early delta initialization if this is supported by the extractor for the DataSource that is called in the source system with this data request. Extractors that support early delta initialization were first supplied with Plug-In (-A) 2002.1.

You cannot simulate an initialization in conjunction with an early delta.

For more details, see the documentation for the data element ROUPDMODE.

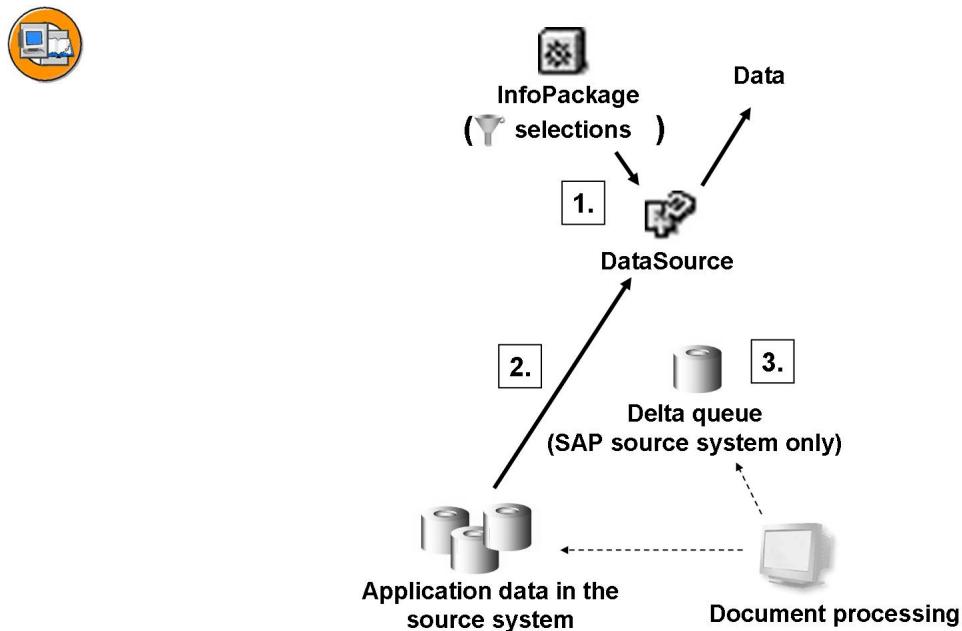


Figure 104: Initializing Delta Process 1

The following process occurs with an initialization run:

1. The delta initialization selection for the DataSource is saved.
2. All the data that corresponds to the selection criteria determined in the InfoPackage is requested (similar to the full update).
3. With request status “green”: The delta queue for the DataSource is generated in the source system.

If a delta queue for the DataSource is being generated in the source system, this means that an empty storage area is being created for new or changed (delta) data records for a DataSource.

In many applications, document processing has to stop when the delta process is being initialized:

- Only after the delta process has been initialized successfully (including generating the delta queue for the DataSource) can you update new or changed (delta) data records to the delta queue. The delta type determines whether or not delta records can be updated directly to the delta queue from the application or the DataSource extractor, however.
- To delimit the initial dataset, if you cannot distinguish between the “initial” data records and the “delta” data records using other information (such as the time stamp) from the DataSource extractor.

If there is no dataset to transfer, you can still initialize the delta process using the “Initialize Without Data Transfer” option.

- Only steps 1 and 3 in the above graphic are executed.
- The control parameters are set in all the necessary tables in the BI and source systems, however, no data is loaded from the source system.
- A message appears in the monitor informing you that a record was loaded. The init simulation request appears in all the data targets where updates occurred. It only contains status entries, however. No data is updated to the data targets in the simulation.

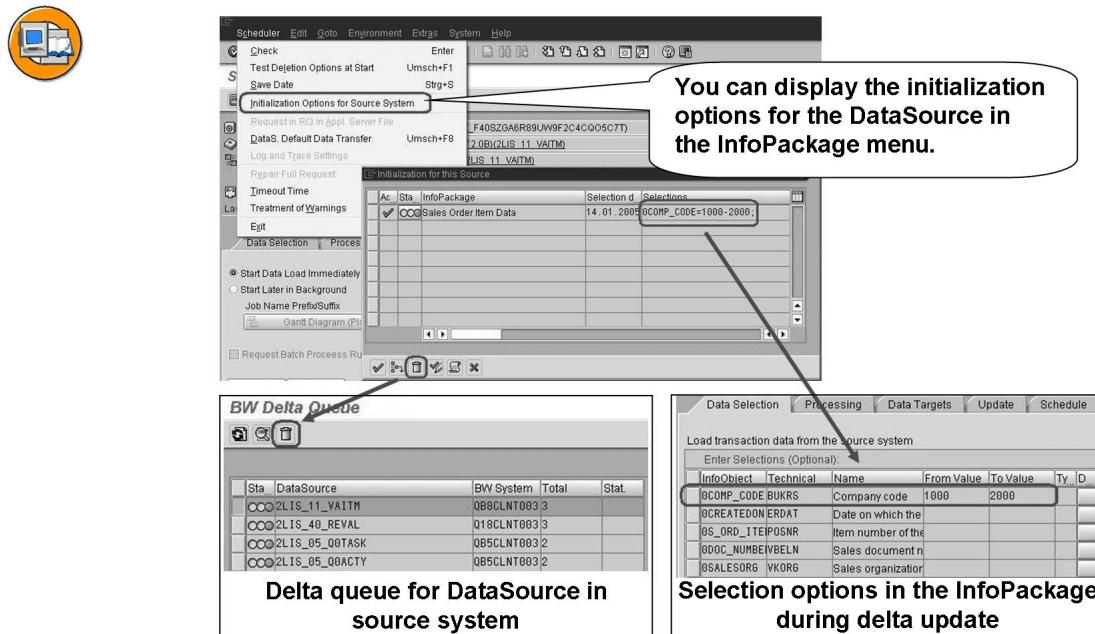


Figure 105: Initializing Delta Process 2

The initialization selections are copied automatically for subsequent delta updates. No other selections can be specified in the delta update!

When you delete all the initialization selections for the DataSource, you delete the delta queue for the DataSource in the source system at the same time!

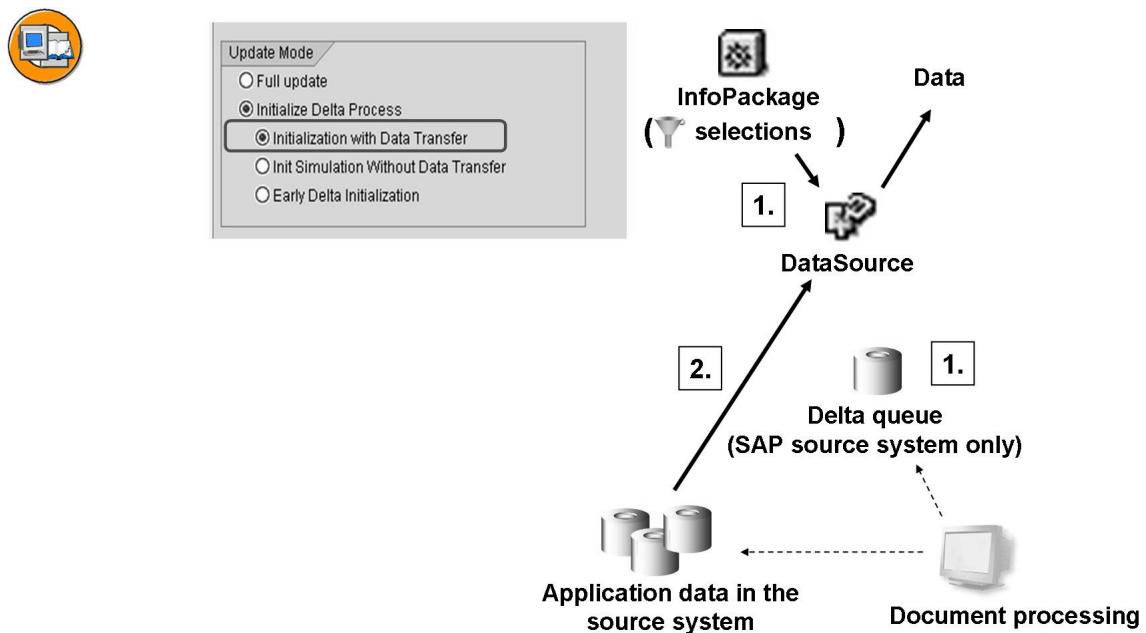


Figure 106: Initializing the Delta Process: Early Delta

If you carry out an early delta initialization, the process is as follows:

1. The selection for the delta initialization of the DataSource is saved and the delta queue for the DataSource is generated in the source system.
2. All the data that corresponds to the selection criteria determined in the InfoPackage is requested (similar to the full update).

The steps “Generate Delta Queue” and “Request Data” are reversed here in comparison with the “normal” way in which the delta process is initialized!

Use the “Early Delta Initialization” indicator in the InfoPackage if you want to avoid a period when no updates can be made (no document processing and update) in the source system while the delta process is being initialized.

Note that in some applications (such as LO Cockpit), an update-free period is nevertheless needed to fill the setup tables. In such cases, the update-free time is merely reduced.

If you want to carry out early delta initialization, the update in the source system can continue (A) and the data can be written to the delta queue while the initialization requests are being processed.



Note: DataSources that support early delta initialization are available as of the release Plug-In 2002.1. The field ZDD_ABLE (characteristic value = ‘X’) for the table *RSOLTPSOURCE* (in BI) or *ROOSOURCE* (in the source system) indicates whether a DataSource can support early delta initialization.

The process described there is based on using a physical system copy that is brought in to initialize the delta process.

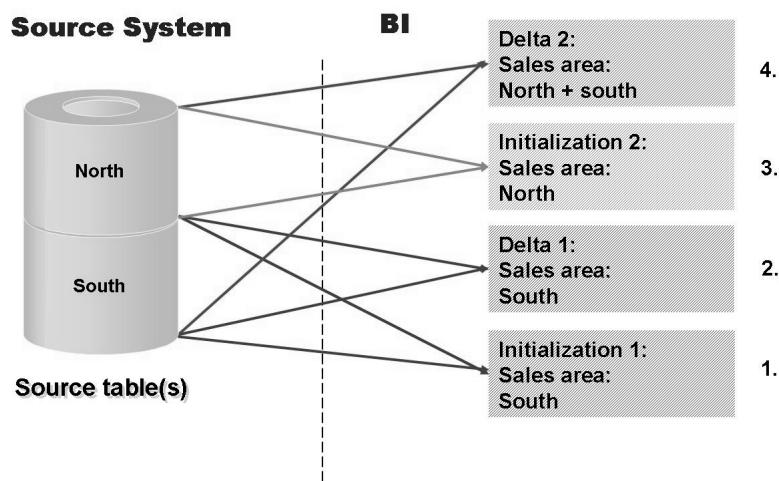


Figure 107: Initializing the Delta Process Step-by-Step

Due to the potentially large volume of data involved, you can initialize the delta process step-by-step. The selections for numerous initializations must not overlap.

If you use a time-based selection when initializing the delta process, you should use an upper maximum time interval (to-date: 12.31.9999). This avoids data records from the delta queue mistakenly not being transferred to BI with future delta updates.



Static historical data can be transferred using the full update function prior to initialization.

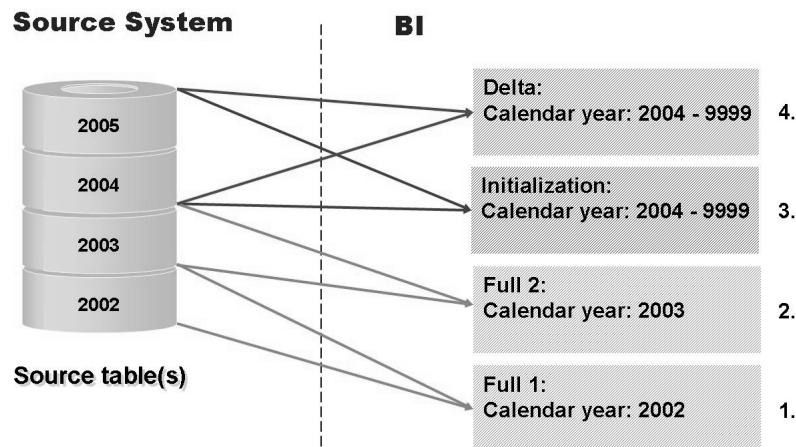


Figure 108: Delta Process Initialization and Full Update

You can reduce the volume of data in the delta process initialization (and thus reduce the processing time) by transferring historical data beforehand by full update.

Historical data in the source system is considered to be static because no further changes are made to it.

Prerequisite: The data extracted by the full update method must be static, in other words, there can be no changes made to these data records at a later date.

Additional tips:

You can use several InfoPackages (with different selections) to make the prior full update transfer in parallel.

If you use a time-based selection when initializing the delta process, you should use an upper maximum time interval (to-date: 12.31.9999). This avoids data records from the delta queue mistakenly not being transferred to BI with future delta updates.

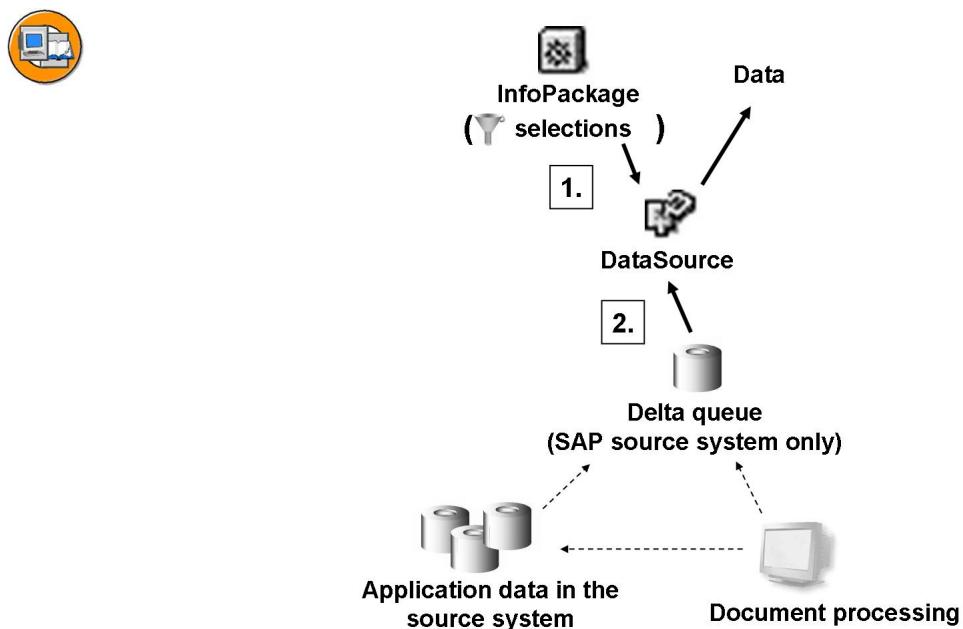


Figure 109: Delta Update

A delta update requests only those data records that were created or changed (or deleted) since the last load for the DataSource in the source system. The delta queue is the central temporary storage location for these delta data records.

The properties of the delta process for the DataSource determine how the delta data records arrive in the delta queue (delta type) and how the changes are recorded (record mode and serialization). Details about these aspects are provided under “Properties of Delta Processes”.

Process flow:

- When you carry out a delta update, the initialization selection for the DataSource is used automatically.
- The new and changed data records are transferred from the delta queue to the BI system.

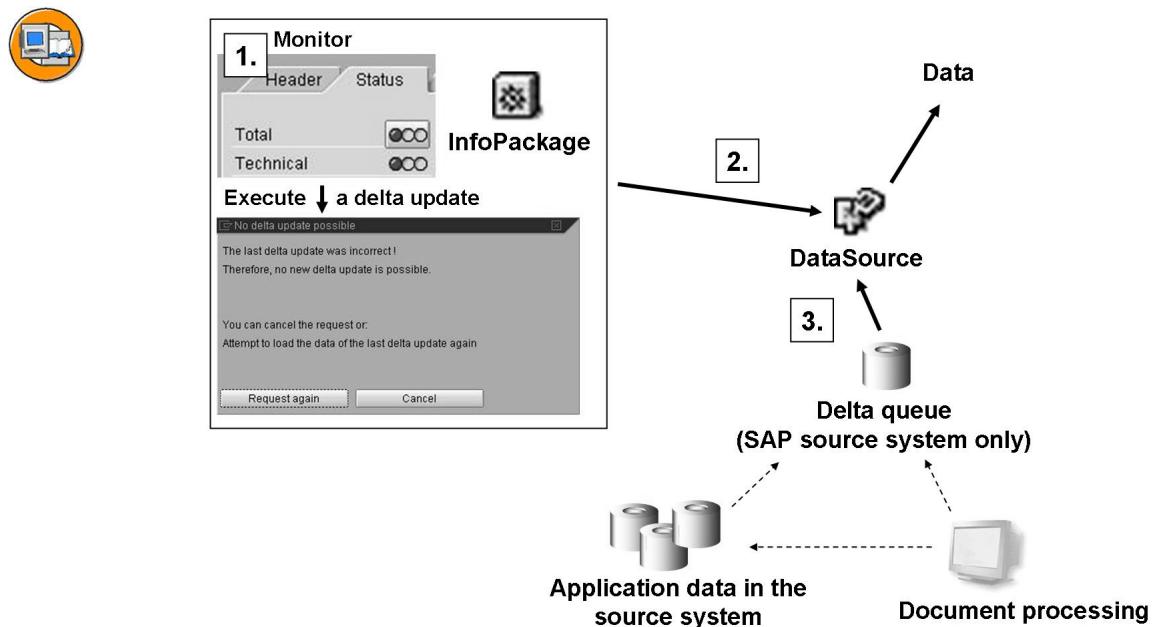


Figure 110: Repeat Delta Update

If a data loading process (request) fails and a red traffic light appears in the BI monitor (transaction RSMO), the previous delta transfer is repeated. In this case, the delta for the same DataSource is requested in the source system again and the source system re-sends the records from the delta queue.

Process flow:

- A delta update fails. When the next delta update is executed, the user is informed that the delta update is repeated when the InfoPackage is executed (this occurs automatically with background processing).
- A data request for update mode R is sent to the source system.
- The new and changed data records are transferred from the delta queue to the BI system again.

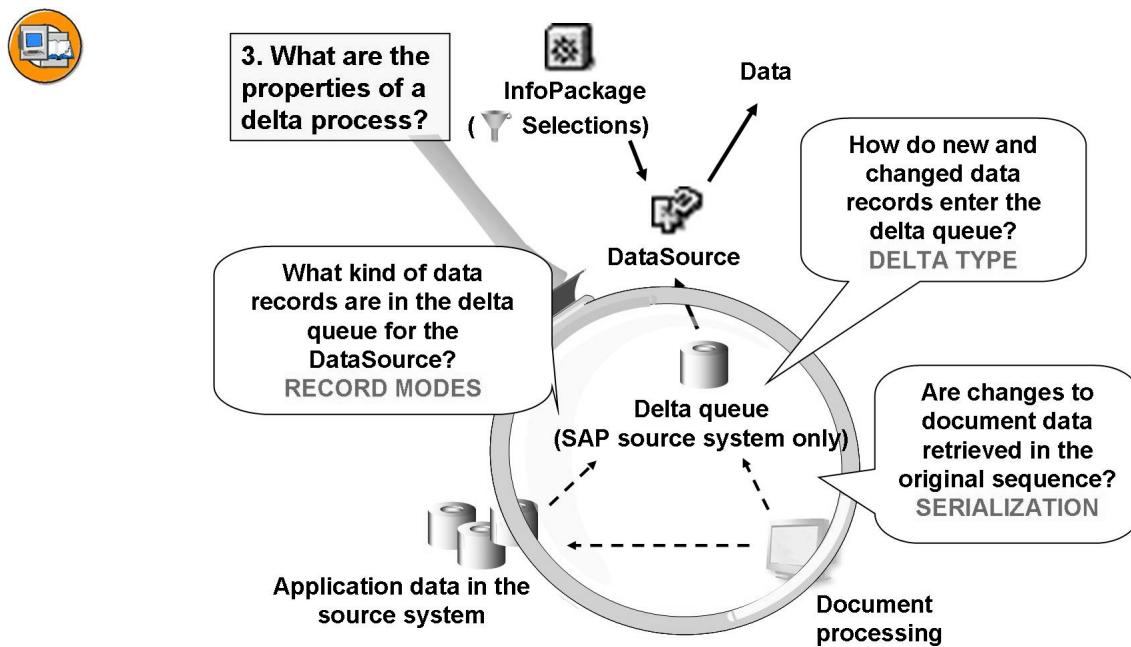


Figure 111: Properties of a Delta Process

The delta process is a property of the DataSource. It indicates how data is transferred. As an attribute of the DataSource, it indicates how the data in the DataSource will be transferred to the data target. This enables you to derive the data targets for which a DataSource is suitable, how to update them, and how the serialization has been done, for example. This information is required when modeling a data flow as well as for managing the BI system.



Data Browser: Table ROOSOURCE

DLTPSOURCE	OBJVERS	TYPE	DELTA
0FI_GL_4	A	TRAN	AIE
2LIS_11_VAITM	A	TRAN	APR

Data Browser: Table RODELTAM Select Entries 20

DELTA	UPDM_NIM	UPDM_BIM	UPDM_AIM	UPDM_MID	UPDM_DEL	UPDM_RIM	DREQSER	DELTATYPE	TXTLG
A							1		Delta only via Full Upload (ODS or InfoPackage Selection)
ABR	X	X	X				2	A	ALE Update Pointer (Master Data)
ABR1	X	X	X				2	D	Compl. Delta w. Delet. Ind. Using Delta Queue (Cube Compat.)
ADD				X		X	1	D	As 'ABR' Process But Only in Series by Request
ADD000				X			1	E	Additive Extraction Using Extractor (e.g. LIS-Infostuct.)
AIE							2	E	No 'ADD' but via Delta Queue (Cube Enabled)
FILED					X		2	E	After-Images with Delete Indicator via Extractor (FI-GL/AP/AR)
AIM					X		2	D	After-Images using Delta Queue (for example, FI-AP/AR)
AIMD					X		2	D	After-Images with Deletion ID Using Delta Queue (btB)
CUBE					X		0	E	InfoCube Extraction
D							2	D	Unspecified Delta using Delta Queue (Not ODS-Capable)
E							2	E	Unspecified Delta via Extractor (non ODS-Capable)
FIL0							2	F	Delta using File Import with After-Images
FIL1							2	F	Delta using File Import with Delta-Images
NEWD	X				X		0	D	Only New Records (Inserts) via Delta Queue (Cube Enabled)
NEWE	X				X		0	E	Only New Records (Inserts) via Extractor (Cube Enabled)
O		X	X	X			1	E	ODS Extraction
ODS	X	X	X	X			2	X	Delta Unspecified (do not use!)
X									

Figure 112: Properties of a Delta Process: RODELTAM

Procedure

- Determine the delta process for the DataSource in the table *ROOSOURCE* (in the source system) or in the table *RSOLTPSOURCE* (in BI for DataSources 3.x) or in the table *RSDS* (in BI for DataSources) respectively
- Determine the properties of the delta process in the table *RODELTAM* (in BI or in the source system)

The table *RODELTAM* defines the properties of the delta process for a DataSource:

- Delta type (explained in detail later)
- Record modes in the delta process (explained in detail later)
- Serialization

The serialization field (*RODELTAM-DREQSER*) describes to what extent the sequence of individual data records within a delta process reflects the origination of these data records in the application. Serialization is above all necessary in overwriting data targets (DataStore object) to enter changes to data records in the correct sequence.

The following provides a complete overview of all the possible characteristic values:

1. No serialization takes place.
2. Serialization required between requests. The data packages within a request do not need to be serialized.
3. Serialization required at data package level. The data packages in a request must be serialized too. This is necessary if an extractor can only supply after images (see record modes) and several records with the same key can be sent within a request, for example.

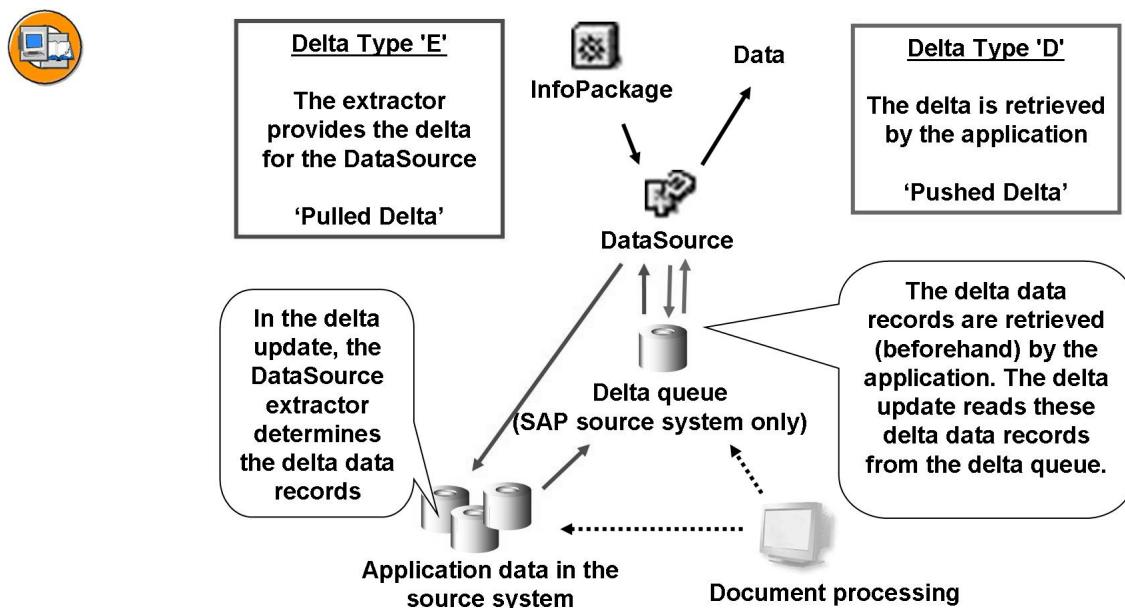


Figure 113: Delta Process: Delta Type

The delta type field (*RODELTAM-DELTATYPE*) is a property of a delta process. It describes how the delta is staged within a delta process. Two of the most important characteristic values that are only used in conjunction with SAP source systems are illustrated in the above graphics.

The following characteristic values are possible:

- ''': The delta type is not defined.
- 'A': The DataSource determines the delta with ALE update pointers. This method is used in the main in connection with DataSources for attributes and texts from SAP source systems. It can also be used to enable generic DataSources to provide delta figures (providing the underlying master data table supports this). ALE update pointers for master data have been available for many years. In the past, they were used to coordinate master data beyond SAP systems.
- 'D': The SAP application writes delta data records directly to the delta queue ("PUSH") for the DataSource. Each data record is either a) stored in the delta queue individually on saving / updating the corresponding transactions in the application (for example, FI-AR/AP or direct delta in the LO Cockpit), or b) written in groups of delta data records (after updating the transaction) to the delta queue by means of application-specific jobs. In each case, however, the delta data records are in the delta queue for the SAP source system before the delta update is executed. In the case of a delta update for the DataSource, this delta queue is read and the data records that exist there are transferred to the BI system. This delta type is normally used in applications in which delta data records cannot be determined through fields or control tables for the underlying application tables.
- 'E': The DataSource determines the delta through the extractor on request. This means that the extractor must be capable of providing the delta records for the DataSource on request ("PULL"). The delta data records that have been determined are placed in the delta queue by the extractor and transferred from there to the requesting BI target system. If you have carried out delta initializations for more than one BI system, or several delta initializations for the same BI system, the extractor has to stage delta records for all the delta initializations and store them in all the delta queues that are specific to the BI target systems. This delta type is normally used in applications in which delta data records can be determined through fields or control tables for the underlying application tables (for example, time stamp information for each data record).
- 'F': The delta data records are loaded by flat file. This delta type is only used for DataSources for flat file source systems. Flat files are imported to the PSA when you execute the InfoPackage. The staging process then begins. The delta queue is not used here.

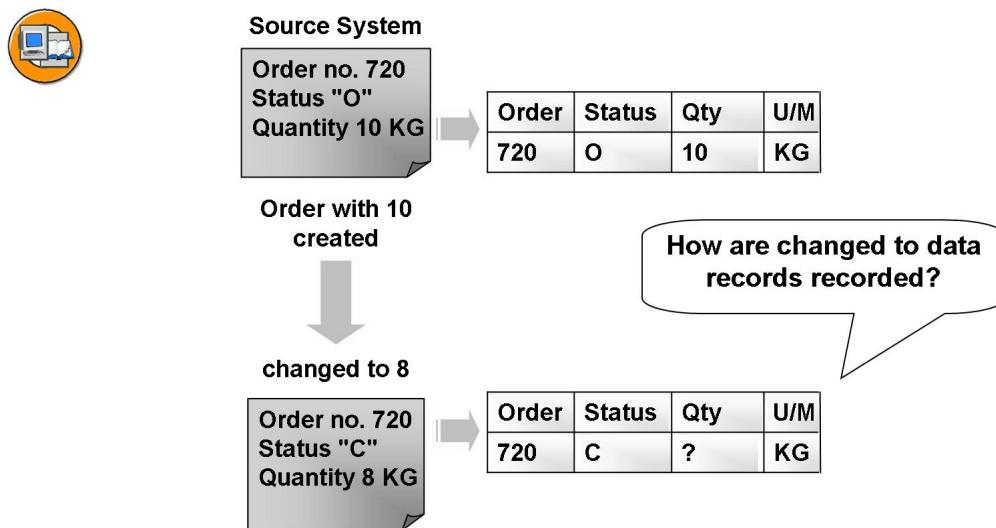


Figure 114: Delta Process: Record Modes 1

The record mode concept deals with the question of how changes to data records are recorded in the delta process. This is exemplified in the following graphics. In this example, the record mode answers the following question: How does the DataSource send data when the quantity in an order has been changed from 10 to 8? You will also learn how the BI system recognizes the type of records transferred.

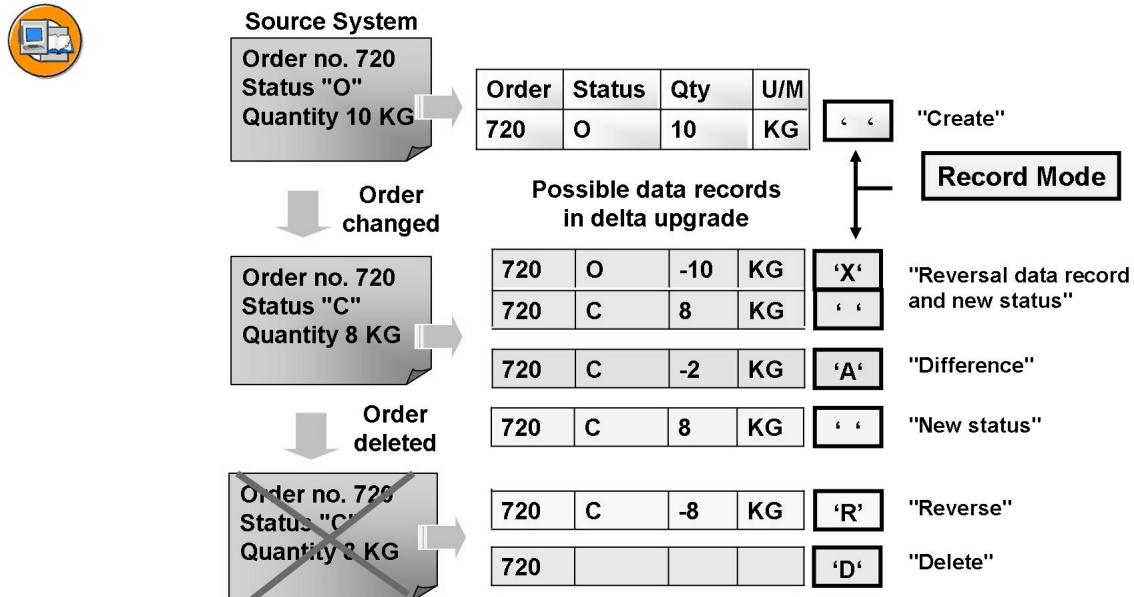


Figure 115: Delta Process: Record Modes 2

The above example illustrates how the order is changed. First, the quantity is changed from 10 to 8 and then the order is deleted in full. It is clear that on creating the order, a new data record is generated when it is saved. It is not clear

how the change in the quantity from 10 to 8 is documented. There are three options for the data that is transferred to the BI system and document this change. They are illustrated in the middle of the graphic.

When the order is deleted in the source system, the BI system can be informed about this in several ways. It can be informed that the record is to be deleted, or it can receive a record with inverse values (see the above example: -10).

The record mode now describes the type of change that a data record contains. The difference between the various delta processes is that each supports only a subset of the seven possible characteristic values. If a DataSource implements a delta process that uses several characteristic values, the field *ROCANCEL* (in which the record mode is saved) is automatically part of the DataSource. This field for the DataSource is assigned to the InfoObject 0RECORDMODE in the BI system.

The characteristic values are as follows:

- ' ': The record provides an after image.

The status of the record is transferred after it has been changed, or after data has been added. The record can only be directly updated to an InfoCube if the corresponding before image is in the request (this will be explained later).

- 'X': The record provides a before image.

The status of the record is transferred before it has been changed or deleted. All attributes for the record that can be aggregated (key figures) must be transferred with a reversed plus/minus sign. Responsibility for reversing the plus/minus sign lies either with the extractor (default) or with the service API. These records are ignored in a non-additive (overwriting) update of a DataStore object. The before image complements the after image.

- 'A': The record provides an additive image.

With attributes that can be aggregated, only the changes are transferred. In the case of attributes that cannot be aggregated, the status after data was changed or created is transferred. The record can be updated to an InfoCube without restrictions, but requires an additive update to be made to a DataStore object.

- 'D': The record must be deleted.

Only the key is transferred. This record (and therefore the DataSource too) can only be updated to a DataStore object.

- 'R': The record provides a reverse image.

The content of this record is equivalent to a before image. The only difference occurs when updating a DataStore object: An existing record with the same key is deleted.

- 'N': The record provides a new image.

The content of this record is equivalent to an after image without a before image. A new image should be transferred instead of an after image when a record is created. The new image complements the reverse image.

In the table *RODELTAM* you determine the characteristic values used by a delta update (columns UPDM_NIM, UPDM_BIM, UPDM_AIM, UPDM_ADD UPDM_DEL and UPDM_RIM). The table must ensure that only meaningful combinations of the characteristic values mentioned above are used in a delta process. A DataSource that uses a delta process can only provide characteristic values in the extracted records for the record mode that are specified in the delta process when you extract data in “delta update” mode.

For more details, see the documentation for the data element *RODMUPDMOD*.

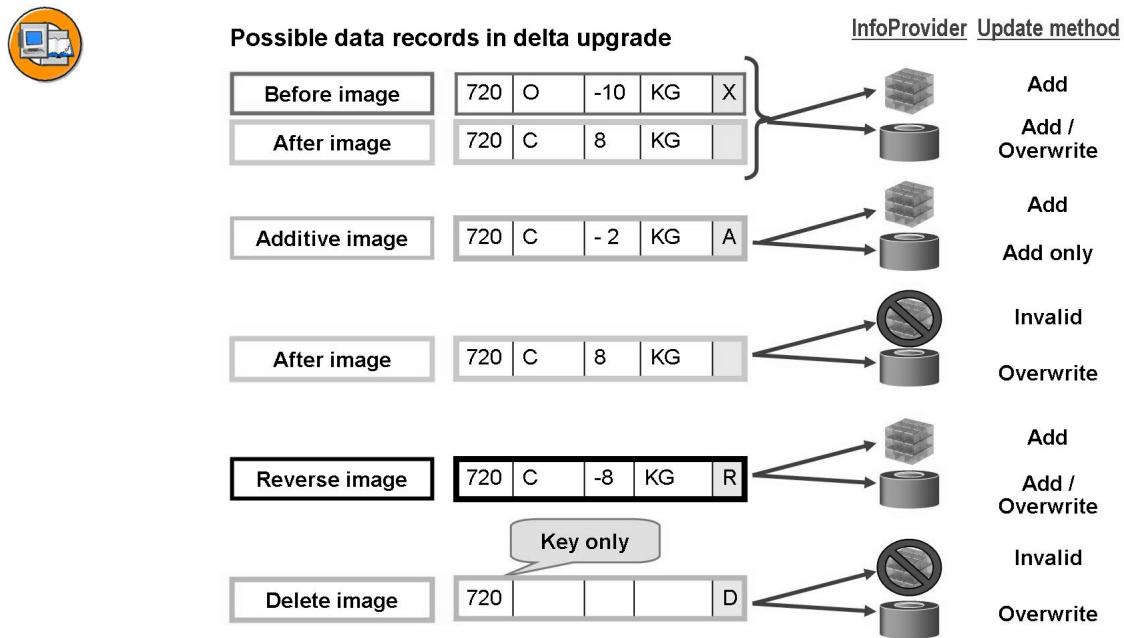


Figure 116: Record Modes and InfoProviders

The graphic describes the connection between the record mode and the possible update to data targets for transaction data.

In an InfoCube only the additive update option is possible. This means that in an InfoCube, data records with the same characteristic value are cumulated but not overwritten.

You can determine the update type for each key figure in the update rules for a DataStore object. It is possible to update a key figure in “add” or “overwrite” mode (with reference to the DataStore object key).

Note the following dependencies when using DataSources capable of providing delta figures in the modeling of a data flow:

- If the DataSource sends both the **before image** and the **after image**, this combination can be loaded to any InfoCube or DataStore object. If the overwrite data setting was made for DataStore objects, only the after image (the last image) arrives in the activation queue table of the DataStore object. If settings are made in the DataStore object so that data is added, both the before and the after image are necessary to load the data correctly to the target.
- If the data that fills the BI system is an **additive image**, the data can be written to an InfoCube or a DataStore object. With a DataStore object, the update type for key figures must be set to “add” and not “overwrite”, however.
- If the DataSource only sends the **after image**, this must first be updated to a DataStore object that is in “overwrite” mode.
- **Reverse images** can be processed by all targets.
- Delete images can only be processed by a DataStore object. InfoCubes cannot process deletions.

Exercise 8: Delta Management: Delta Process

Exercise Objectives

After completing this exercise, you will be able to:

- Explain the underlying control tables in DataSource management in the SAP source system
- Interpret the content of these tables in terms of their effect on data modeling and the management of your BI system

Business Example

The purpose of this exercise is to enable the participants to understand the underlying tables in the delta management of DataSources.



Note: This exercise serves to refresh the knowledge you have acquired so far. Unfortunately, only general questions regarding delta management can be dealt with here. In view of the large number of Business Content DataSources, questions specific to Business Content cannot be discussed. Further information on Business Content is available in the documentation for the most up-to-date BI Business Content release and from searching the relevant notes (with the name of the DataSource) in the SAP Service Marketplace.

Task 1:

1. Take a look at the metadata in the table ROOSOURCE in the SAP source system. This table stores general information on DataSources in the SAP source system. Look at the data and the fields in the table and answer the following questions.
2. Which fields are defined as KEY FIELDS in the table?
Why?
3. Now display the content of the table.
4. Display the details for the DataSource **0EC_PCA_1**.
5. Which extraction method has been maintained for 0EC_PCA_1?
6. What is the name of the function module (check in the “Extractor” field)?
7. Does this DataSource support a delta process? If so, which?
8. What is the name of the extraction structure?

Continued on next page

Task 2:

Now use the content of the table ROOSOURCE to determine the delta process for the following DataSources:

2LIS_12_VCITM (Supply Item Data)

0FI_AR_4 (Customers: Line Items via Delta Extraction)

0HR_PY_1 (Payroll Data)

1. Exit the display of the entries in the table ROOSOURCE (see above) and return to the selection screen for the contents of the table ROOSOURCE (Data Browser: Table ROOSOURCE: Selection screen).
2. Display the entries in the table ROOSOURCE for the DataSources listed above (2LIS_12_VCITM, 0FI_AR_4 and 0HR_PY_1). Enter the name of the DataSource(s) in the field OLTPSOURCE (DataSource).
3. Note down the delta process for the corresponding DataSources in the table you will find under point 4 of this exercise.

Task 3:

1. Take a look at the metadata table RODELTAM in the BI system. This table provides detailed information about the various types of delta process. The table exists in both the BI and the OLTP system. For the purpose of this exercise, use the table in BI.
2. Which fields are defined as KEY FIELDS in the table?
Why?
3. Now display the content of the table.
4. Take a look at the property details for the delta process determined above.
5. Look at the content of the *DELTATYPE* field for the three delta processes determined above. Does the application write the data to the data queue, or is this done by the extractor for the DataSource as soon as the corresponding InfoPackage has been executed? In the table under task 4, enter a cross for the correct answer for each DataSource and its delta process (in column 3 or 4).
6. Which record types are supported by the respective delta processes? What implications does this have for possible updates of these DataSources to DataStore objects (update type “addition” or “overwrite”) and to InfoCubes? Place a cross by the possible options for each DataSource and its delta process in columns 5, 6 and 7 of the table that appears under task 4.

Continued on next page

Task 4:

Enter your observations from tasks 2 and 3 in the following table:

1.

Data-Source	Delta process	The delta is staged by the application	The extractor provides the delta for the Data-Source	DataStore object update type		In-foCube
				Addi-tion	Over-write	
2LIS_- 12_VC- ITM						
0FI_- AR_4						
0HR_- PY_1						

Solution 8: Delta Management: Delta Process

Task 1:

1. Take a look at the metadata in the table ROOSOURCE in the SAP source system. This table stores general information on DataSources in the SAP source system. Look at the data and the fields in the table and answer the following questions.

- a) *BI Administration → Data Warehousing Workbench → Source Systems → Context Menu (right mouse click) on Source System 'T90CLNT090' → Customizing Extractors.*

Enter the transaction code /OSE12. → Enter the table name *ROOSOURCE* in database table input field. Click on the *DISPLAY* symbol towards the bottom of the screen.

2. Which fields are defined as KEY FIELDS in the table?

Why?

- a) “DataSource” and “version of an OLTP Source”

This is the central table that describes the metadata for a DataSource. The DataSource field is therefore the key for the table.

SAP also provides the option of saving different versions of the DataSource:

- “Delivered” (Business Content)
- “Active” (Live version)
- “Modified” (in processing)

3. Now display the content of the table.

- a) Choose the *Content* symbol 

Click on the *Execute* symbol  to see the contents of the table.

4. Display the details for the DataSource **0EC_PCA_1**.

- a) Scroll down until you find the DataSource *0EC_PCA_1*. Double-click on the DataSource.

Note: In case you cannot find the DataSource, check to make sure that you are in the SAP system that is connected to BI. The table ROOSOURCE exists in the BI system too. Here, however, it contains the DataSources for the DataMarts.

Continued on next page

5. Which extraction method has been maintained for 0EC_PCA_1?
 - a) F1 = function module (complete interface)
 6. What is the name of the function module (check in the “Extractor” field)?
 - a) ECPCA_BIW_GET_DATA
→ This is the DataSource extractor.
- Note: To display the source code for the function module, enter transaction SE37.
7. Does this DataSource support a delta process? If so, which?
 - a) Yes, it supports the delta process. Every entry except for " " means that a delta process is supported. In this case the delta process “ADDD” is supported.
 8. What is the name of the extraction structure?
 - a) ISPCACST

Task 2:

Now use the content of the table ROOSOURCE to determine the delta process for the following DataSources:

2LIS_12_VCITM (Supply Item Data)

0FI_AR_4 (Customers: Line Items via Delta Extraction)

0HR_PY_1 (Payroll Data)

1. Exit the display of the entries in the table ROOSOURCE (see above) and return to the selection screen for the contents of the table ROOSOURCE (Data Browser: Table ROOSOURCE: Selection screen).
 - a) Click twice on the *Back* symbol to return to the Data Browser for the table ROOSOURCE.
(Alternatively, you can enter the transaction code /NSE16 and enter the table “ROOSOURCE” in the field *Table Name*. Then choose the *Content* symbol

Continued on next page

2. Display the entries in the table ROOSOURCE for the DataSources listed above (2LIS_12_VCITM, 0FI_AR_4 and 0HR_PY_1). Enter the name of the DataSource(s) in the field OLTPSOURCE (DataSource).
 - a) Enter the name of the corresponding DataSource(s) in the field OLTPSOURCE (DataSource).

Click on the *Execute* symbol  to see the contents of the table.
Alternatively, you can also use the search function.
3. Note down the delta process for the corresponding DataSources in the table you will find under point 4 of this exercise.
 - a) You can find the technical name of the delta process for a DataSource in the field *DELTA* of the display. Note down this delta process in the table under point 4.

Task 3:

1. Take a look at the metadata table RODELTAM in the BI system. This table provides detailed information about the various types of delta process. The table exists in both the BI and the OLTP system. For the purpose of this exercise, use the table in BI.
 - a) Path:
Enter the transaction code /OSE12. → Enter the table *RODELTAM* in the database table input field and then click on the *DISPLAY* symbol lower down the screen.
2. Which fields are defined as KEY FIELDS in the table?
Why?
 - a) DELTA delta process for a DataSource
This table contains details for the respective delta process. These are:
 - Delta type
 - Record mode used
 - Serialization
3. Now display the content of the table.
 - a) Choose the *Content* symbol 

Click on the *Execute* symbol  to see the contents of the table.

Continued on next page

4. Take a look at the property details for the delta process determined above.
 - a) Reminder: The delta process “ABR” for the DataSource 2LIS_12_VCITM, the delta process “AIE” for the DataSource 0FI_AR_4 and the delta process “ADD” for the DataSource 0HR_PY_1 were determined above.
5. Look at the content of the *DELTATYPE* field for the three delta processes determined above. Does the application write the data to the data queue, or is this done by the extractor for the DataSource as soon as the corresponding InfoPackage has been executed? In the table under task 4, enter a cross for the correct answer for each DataSource and its delta process (in column 3 or 4).
 - a) If the content of the Delta Type for a DataSource field is “D”, this means that the application writes the changed data records to the delta queue before they are collected by BI when a delta update for a DataSource occurs.→ “Delta is staged by the application” must be set.
If the content of the Delta Type for a DataSource field is “E”, this means that the DataSource extractor determines the changed records during the delta update extraction. The changed records are placed in the delta queue by the extractor and transferred from there to the BI system.→ “Extractor provides the delta for the DataSource” must be set.

Continued on next page

6. Which record types are supported by the respective delta processes? What implications does this have for possible updates of these DataSources to DataStore objects (update type “addition” or “overwrite”) and to InfoCubes? Place a cross by the possible options for each DataSource and its delta process in columns 5, 6 and 7 of the table that appears under task 4.
- a) Delta process “ABR”
- “After image” (UPDM_AIM), “before image” (UPDM_BIM), “reverse images” (UPDM_RIM) and “new images” (UPDM_NIM).
- Since this DataSource transfers both after and before images in a delta update, the data can be updated to both a DataStore object (update types “addition” and “overwrite” are possible) and an InfoCube.
- Delta process “AIE”
- only “after image” (UPDM_AIM).
- Since only after images are transferred in a delta update, this data can only be loaded to a DataStore object with the “overwrite” update type.
- Delta process “ADD”
- only “additive image” (UPDM_ADD).
- Since only additive images are transferred in a delta update, this data can be updated to DataStore objects (update type “addition”) and InfoCubes.

Task 4:

Enter your observations from tasks 2 and 3 in the following table:

1.

Continued on next page

Data-Source	Delta process	The delta is staged by the application	The extractor provides the delta for the Data-Source	DataStore object update type		In-foCube
				Addi-tion	Over-write	
2LIS_-12_VC-ITM						
0FI_-AR_4						
0HR_-PY_1						

- a) After you have carried out steps 2 and 3 in this exercise, the solution table should be filled as follows:

Data-Source	Delta process	The delta is staged by the application	The extractor provides the delta for the Data-Source	DataStore object update type		In-foCube
				Addi-tion	Over-write	
2LIS_-12_VC-ITM	ABR	✓		✓	✓	✓
0FI_-AR_4	AIE		✓		✓	
0HR_-PY_1	ADD		✓	✓		✓



Lesson Summary

You should now be able to:

- Explain the basic concepts of delta management and describe how it works

Lesson: Examples from Applications and Source Systems

Lesson Overview

In this lesson, we will explain delta management in the context of examples from SAP applications. This serves as the basis for explaining and understanding the delta process from other application areas.



Lesson Objectives

After completing this lesson, you will be able to:

- Provide examples of how delta management works

Business Example

Your company has recognized the need to integrate people, information and business processes in a heterogeneous system landscape and would like to benefit from it. Practice has shown, though, that loading very large datasets puts a strain on hardware and system performance.

For this reason, it is necessary to examine if and how the data records can be loaded into BI with a delta process. It is important to understand the modes of operation and the different variants of a delta loading process.

Examples from Applications and Source Systems

The figure shows a comparison of two different Business Content DataSources that have different delta processes (they are similar only with regard to serialization). Both delta processes provide the data records in the same sequence in which they were created in the source system.

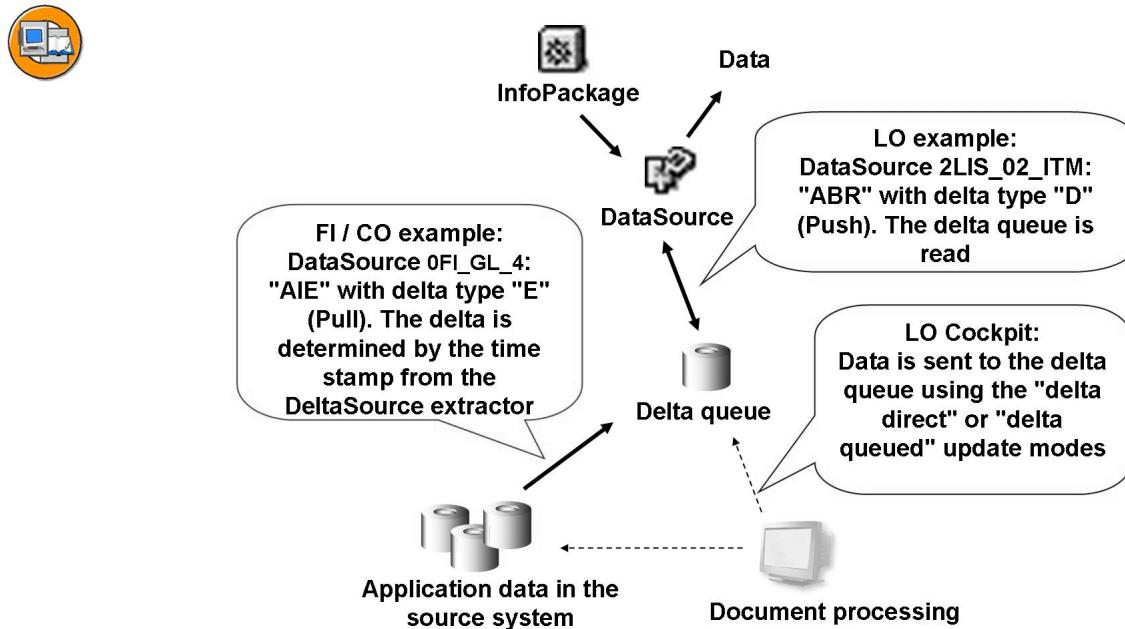


Figure 117: Examples of Delta Processes in Business Content

Here is a short comparison:

DataSource 2LIS_02_ITM (purchasing data (item level)) has the delta process "ABR" with delta type "D" (push):

- Because this is delta type "push", the delta data records from the application are posted to the delta queue before they are extracted from the delta queue as part of a delta update. If the DataSources are from the LO Cockpit, this "push" takes place with the methods described there: "Direct delta" (direct from the application, V1 – updating) or "queued delta" (from the application to the extraction queue (V1) and from there to the delta queue with a collective run / job).
- The DataSource continues to provide after, before and reverse images. For this reason, you can use both InfoCubes and DataStore objects as InfoProviders with no restrictions.

DataSource 0FI_GL_4 (General ledger: Line items via delta extraction) has the delta process "AIE" with delta type "E" (pull):

- Because this is a delta type "pull", the delta data records are determined during the delta update by the DataSource extractor, updated to the delta queue and passed on to BI directly from there.
- The DataSource only returns the after images. Direct updating into InfoCubes is not possible for this Business Content DataSource. First, a DataStore object in the data flow has to be addressed that updates key figures with the update type "overwrite" to the update rules.

Important!

A delta process does not indicate **HOW** the delta is determined in the application. This question is resolved in different technical ways in the DataSource extractor or in the application itself and depends on the environment and the data model of the specific application. In our example, the DataSource OFI_GL_4 determines its delta based on a time stamp field in the underlying source table (the changed data records can be selected in the extraction by the extractor of the DataSource using the field). The source tables of the DataSource 2LIS_02_ITM do not offer this precondition, however. For this reason, in this DataSource, the delta is “logged” at the moment of the update (for example “save” in the dialog transaction) so that it can be extracted later in a delta update.

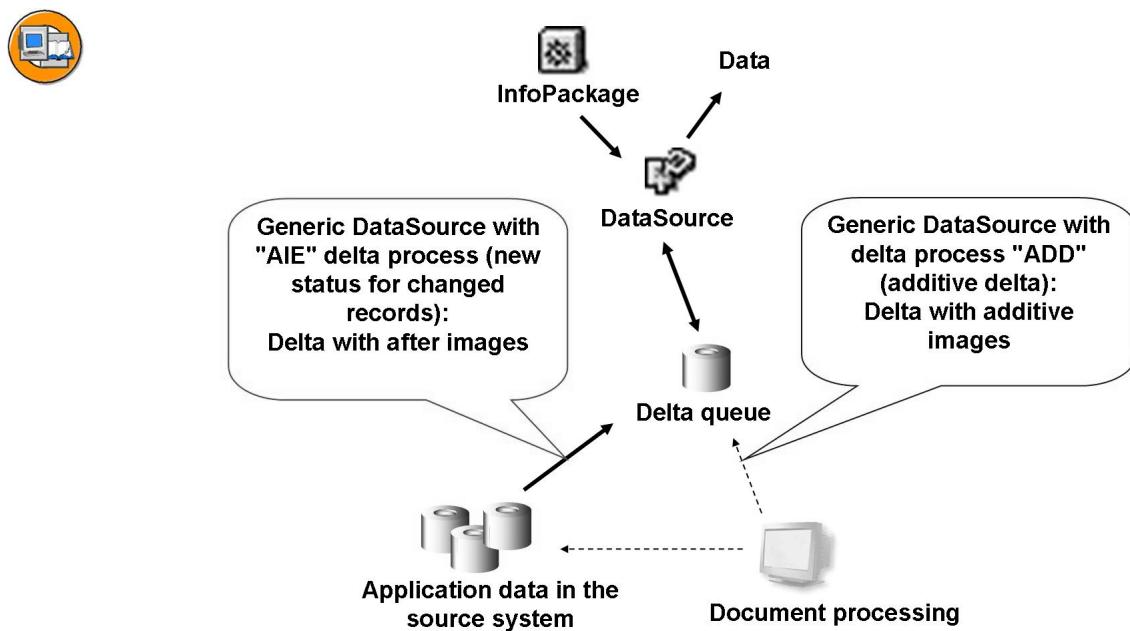


Figure 118: Examples of Delta Processes with a Generic DataSource

With delta processes using generic DataSources, the data is transferred into BI with the delta queue.

- With delta type *additive* delta, the record to be loaded only returns the respective changes to key figures for key figures that can be aggregated. The extracted data is added in BI. DataSources with this delta type can supply DataStore objects and InfoCubes with data. With delta type *additive* delta, the record to be loaded only returns the respective changes to key figures for key figures that can be aggregated. The extracted data is added in BI.
- With delta type *new status for changed records*, each of the records to be loaded returns the new status for all key figures and characteristics. The values in BI are overwritten. DataSources with this delta type can write the data to DataStore objects and master data tables.

Note that you implement a delta process for a UD Connect DataSource with the same tools because, from a technical viewpoint, a UD Connect DataSource is a generic DataSource with a function module for the source system “Myself”.

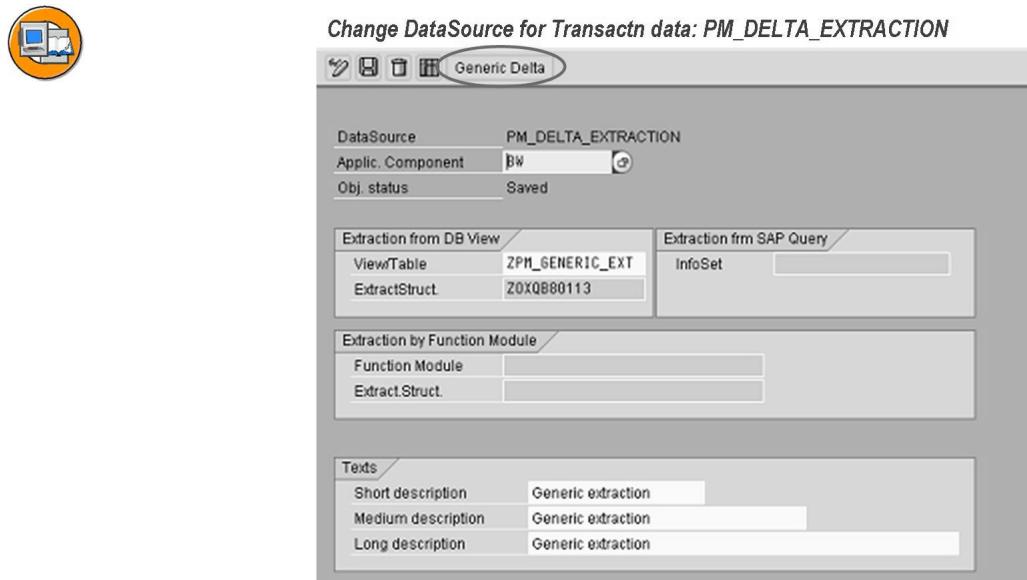
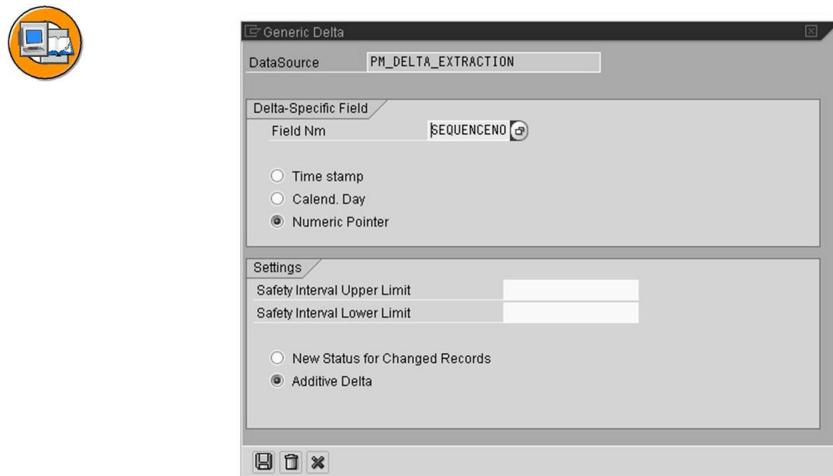


Figure 119: Generic Delta Services: Example 1

You can also integrate a delta process with a “Generic DataSource”, if necessary. However, certain prerequisites need to be fulfilled for this. There must be a way of identifying the data records that have already been transferred in order to be able to determine the delta.



Possible delta extraction characteristics:

- Time stamp
- Calendar day
- Numeric pointer (such as document number, for example)

Figure 120: Generic Delta Services: Example 2

If a field exists in the extraction structure of a DataSource that contains values that increase monotonously over time, you can define delta capability for this DataSource. If this type of delta-relevant field is present in the extraction structure, for example, a time stamp, the volume of data transferred in delta mode is determined by comparing the maximum value transferred during the last load with the data that has accrued since. Only the data that is new is transferred.

To be able to determine the delta, generic delta management converts the update mode into a selection criterion. The selections for the request are expanded by one interval for the delta-relevant field. The lower limit of the interval is known from the last extraction, the upper limit is determined by the current value, for example, the time stamp at the time of extraction. You can use safety intervals to ensure that all the data is included in the extractions (see below). After the data request was transferred to the extractor and the data extracted, the extractor informs generic delta management that the pointer can be set to the upper limit of the previously returned interval.

Safety Interval Upper Limit of a Delta Selection

This field is used by DataSources that determine their delta generically using a monotonously increasing field in the extraction structure.

It includes the gap between the current highest level at the time of the delta or delta init extraction and the data that was actually read.

If the value is initial, there is the danger that records resulting during extraction could not be extracted.

Example: A time stamp is used to specify the delta. The time stamp that was last read was 12:00:00. The next delta extraction starts at 12:30:00. The selection interval is thus 12:00:00 to 12:30:00. At the end of the extraction, the counter is set to 12:30:00.

A record - such as a document - is created at 12:25 and saved at 12:35. It is not included in the extracted data, but is not extracted the next time round due to its time stamp.

For this reason, the safety interval between read and transferred data should always be larger than the maximum time range that is needed to create a record for this DataSource (for time stamp delta), or should be a sufficiently large data interval (with delta specification using a sequential number).

Safety interval lower limit

This field includes the value that has to be extracted from the highest level of the last delta extraction to determine the lowest value of the time stamp for the following delta extraction.

Example: a time stamp is used to specify the delta. The extracted data is master data: only after images are transferred that overwrite the status in BI. With this type of data, a record can be extracted into BI twice without causing problems.

Taking this fact into consideration, the current time stamp can always be used as the upper limit during extraction: the lower limit of the next extraction then does not connect seamlessly to the upper limit of the last extraction. Instead it accepts a value that corresponds to this upper limit minus a safety interval. This safety interval has to be large enough that all values that already received a time stamp at the time of the last extraction, but were not read, are now part of the extraction. Records may be transferred twice, of course, but this does not affect anything for the reasons stated above.



Note: A safety interval should only be specified for the lower limit with the delta process new status for changed records, in other words, when the status in BI is overwritten. In this case, duplicate data records, which can occur with such a safety interval, have no effect in BI.



Check status of delta queue in RSA7:

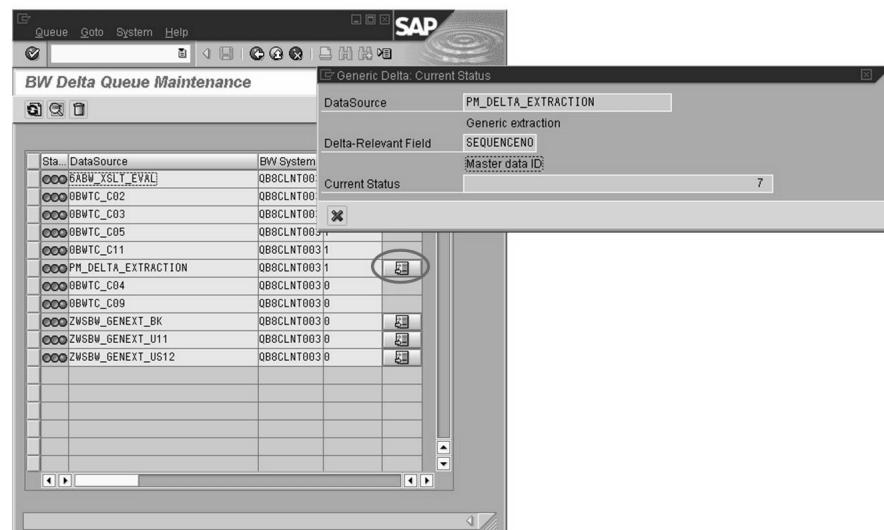


Figure 121: Generic Delta Services: Example 3

With delta processes using flat files, the data is not transferred to BI using the delta queue. Instead it is loaded directly from the DataSource to the PSA. The delta process set in the transfer rule for this Flat File DataSource must be notified of the underlying data and its delta properties.

Example (data target for the flat file DataSource in BI is a DataStore object that has the InfoObject “Document No.” as the only key):

1. First the following data record is loaded into the DataStore object of BI with a Flat File DataSource by initializing the data process.

Document No.: 4711

Material: M-01

Quantity: 12

Unit of measure: ST (piece)

2. Then the following change record is transferred to BI as part of a delta process for the same Flat File DataSource.

Document No.: 4711

Material: M-01

Quantity:15

Unit of measure: ST

3. Depending on the delta process of the Flat File DataSource, the following image results in the DataStore object after updating this delta record:

FIL0: Delta via flat file with after images

Document No.: 4711

Material: M-01

Quantity:15

Unit of measure: ST

FIL0: Delta via flat file with additive images

Document No.: 4711

Material: M-01

Quantity: 27

Unit of measure: ST

If an XML / SOAP DataSource is generated on the basis of a Flat File DataSource, the generated DataSource is capable of producing delta figures. The following dependencies are valid:

- Flat file DataSource: Delta process FIL0 (delta via file import with after images) → XML / SOAP DataSource: Delta process AIM (after images via delta queue)
- Flat File DataSource: Delta process FIL1 (delta via file import with additive images) → XML / SOAP DataSource: Delta process ABR (complete delta with delete indicator via delta queue)
- Flat File DataSource: No delta process → XML / SOAP DataSource: Delta process AIM (after images via delta queue)

Exercise 9: Generic Delta for Transaction Data

Exercise Objectives

After completing this exercise, you will be able to:

- Create generic DataSources for transaction data and make it capable of producing delta figures
- • Create Transformations and Data Transfer Processes to upload data into DataStore Objects
- • Load transaction data in delta mode into the DataStore Object

Business Example

In your company, you want to load data from a user-defined database table into a DataStore object in the Business Information Warehouse. This upload must be delta capable.

Task 1:

Create a generic transaction data DataSource in your BI system and make it delta capable.



Hint: For reasons of simplification, the transaction data DataSource is created in the BI system. The DataSource is created according to the same principle as in the SAP OLTP system.

1. The trainer executes the program **ZT_FILL_TABLE_GEN_EXTRACTION** with the parameter INIT.
2. Check the structure and content of the table **ZTR_GENERIC_EXT** with the transaction SE16:
3. After database table in the BI system is filled, it can be used during the subsequent creation of the new DataSource as an extractor for the BI system. Branch to the BI-IMG (transaction SBIW) and create the new DataSource for transaction data **ZGENEXT_##**.

Define your DataSource as follows:

Field Name	Value
<i>Application component</i>	ZT_BW350_GR##
<i>View/Table</i>	ZTR_GENERIC_EXT

Continued on next page

Field Name	Value
Short Text	Gen. Delta GR##
Medium Text	Generic Delta GR##
Long Text	Generic Delta GR##

Define the delta-capability of your DataSource using the field *SEQUENCENO* (master data ID) as a Numeric Pointer and specify **New status for changed records** as delta mode. Save the DataSource.

Task 2:

After the DataSource has been created, it has to be replicated and the required prerequisites must be created so that the data from the table **ZTR_GENERIC_EXT** can be loaded into the associated DataStore object of BI.

1. In the Data Warehousing Workbench for your BI system, perform a DataSource replication for your application component (ZT_BW350_GR##) in the foreground within your Myself-System “FB4CLNT800”.



Hint: Do not replicate the entire Myself-System!!!

In the dialog box *Unknown DataSource*, accept the default and replicate your DataSource as a BI DataSource (and not as a 3.x DataSource).

Do not forget to activate your BI DataSource **ZGENEXT_##** after replication to BI!

2. You now have to create a Transformation between your DataSource **ZGENEXT_##** and the Data Store-Object **T_GDGR##**. The DataStore Object **T_GDGR##** has already been created in the BI system.

Assign the fields between your DataSource **ZGENEXT_##** and the DataStore Object **T_GDGR##** as follows:

Field Name	Enter
<i>SEQUENCENO</i>	0TCTREQSID
<i>CALDAY</i>	0CALDAY
<i>MATERIAL</i>	0MATERIAL
<i>T_SLSQTY</i>	0CRM_SALQTY
<i>UNIT</i>	0SALES_UNIT

Do not forget to change the aggregation type for Key Figure **0CRM_SALQTY** to **Overwrite**.

Continued on next page

3. Create an InfoPackage for DataSource ZGENEXT_## for use in loading data to the PSA. Use **ZGENEXT_## Delta Initialization** as the InfoPackage description. .

Set the Update Mode on the *Update* tab to **Initialize Delta Process – Initialization with Data Transfer**.

On *Schedule* tab start the data load immediately. Check your data load in the *Data Load Monitor*.

4. To load the data from the PSA table of DataSource **ZGENEXT_##** into the associated DataStore Object **T_GDGR##**, you have to create a Data Transfer Process. On the *Extraction* tab ensure that the *Extraction Mode* is set to **Delta**. Execute the Data Transfer Process.
5. Check the status of the extraction and the contents of the DataStore Object in the *DTP Monitor* respectively within *DataStore Object Administration*.
6. The trainer starts the program **ZT_FILL_TABLE_GEN_EXTRACTION** with the parameter **DELTA** to generate new delta records in the table.



Caution: Since there is only one program, participants should wait until the trainer has performed this step before proceeding.

7. Check the content of the table **ZTR_GENERIC_EXT** after the program has been executed with transaction SE16.
8. Check the delta queue for the DataSource **ZGENEXT_##** and the status of the generic delta using transaction code RSA7.
9. Create a second InfoPackage for your DataSource **ZGENEXT_##** in order to request the delta by loading the new contents of the table **ZTR_GENERIC_EXT** into the PSA table of your DataSource. Use **ZGENEXT_## Delta** as InfoPackage description. Set the Update Mode on the *Update* tab to **Delta Update**. On *Schedule* tab start the data load immediately. Check your data load in the *Data Load Monitor*.
10. Load the delta from the PSA table into your DataStore Object **T_GDGR##** by executing the existing Data Transfer Process again.
11. Check the status of the extraction and the contents of the DataStore Object in the *DTP Monitor* respectively within *DataStore Object Administration*.

Solution 9: Generic Delta for Transaction Data

Task 1:

Create a generic transaction data DataSource in your BI system and make it delta capable.



Hint: For reasons of simplification, the transaction data DataSource is created in the BI system. The DataSource is created according to the same principle as in the SAP OLTP system.

1. The trainer executes the program **ZT_FILL_TABLE_GEN_EXTRACTION** with the parameter INIT.
 - a) The trainer **only** will use transaction SE38 to execute the ABAP program.
 **Caution:** Since there is only one program, participants should wait until the trainer has performed this step before proceeding.
2. Check the structure and content of the table **ZTR_GENERIC_EXT** with the transaction SE16:
 - a) Enter the transaction code /NSE16. Choose the *Content* symbol 
3. After database table in the BI system is filled, it can be used during the subsequent creation of the new DataSource as an extractor for the BI system. Branch to the BI-IMG (transaction SBIW) and create the new DataSource for transaction data **ZGENEXT_##**.

Define your DataSource as follows:

Field Name	Value
<i>Application component</i>	ZT_BW350_GR##
<i>View/Table</i>	ZTR_GENERIC_EXT
<i>Short Text</i>	Gen. Delta GR##
<i>Medium Text</i>	Generic Delta GR##
<i>Long Text</i>	Generic Delta GR##

Continued on next page

Define the delta-capability of your DataSource using the field **SEQUENCENO** (master data ID) as a Numeric Pointer and specify **New status for changed records** as delta mode. Save the DataSource.

- a) *BI System → Transaction: /NSBIW → Generic DataSources → Maintain Generic DataSource*

Choose *Execute* 

Field Name	Enter
Radio Button: <i>Transaction Data</i>	<input checked="" type="checkbox"/>
<i>Transaction data</i>	ZGENEXT_##

Choose *Create* 

Define your DataSource as follows:

Field Name	Value
<i>Application Component</i>	ZT_BW350_GR##
<i>View/Table</i>	ZTR_GENERIC_EXT
<i>Short Text</i>	Gen. Delta GR##
<i>Medium Text</i>	Generic Delta GR##
<i>Long Text</i>	Generic Delta GR##

Button Generic Delta → Delta-Specific Field: Numeric Pointer;

Field Name	Value
<i>Field Name</i>	SEQUENCENO

Settings: **New Status for Changed Records**

Choose *Save* 

Choose *Local Object*

Choose *Save* 

Choose *Back* 

Continued on next page

Task 2:

After the DataSource has been created, it has to be replicated and the required prerequisites must be created so that the data from the table **ZTR_GENERIC_EXT** can be loaded into the associated DataStore object of BI.

1. In the Data Warehousing Workbench for your BI system, perform a DataSource replication for your application component (ZT_BW350_GR##) in the foreground within your Myself-System “FB4CLNT800”.



Hint: Do not replicate the entire Myself-System!!!

In the dialog box *Unknown DataSource*, accept the default and replicate your DataSource as a BI DataSource (and not as a 3.x DataSource).

Do not forget to activate your BI DataSource **ZGENEXT_##** after replication to BI!

- a) *Data Warehousing Workbench → Modeling → Source Systems Context Menu for FB4CLNT800: DataSource Overview*
 - b) *OBW_DATASOURCES → ZBW_TRAINING → ZT_BW350 → Context Menu for the Application Component ZT_BW350_GR## → Choose Replicate Metadata.*
 - c) In the dialog box *Unknown DataSource*, accept the default and choose the Continue icon.
→ **Note:** This will create a BI DataSource rather than a DataSource 3.x.
 - d) From the context menu of your DataSource **ZGENEXT##** choose *Change* and when your DataSource appears in change mode choose the Activate icon.
2. You now have to create a Transformation between your DataSource **ZGENEXT_##** and the Data Store-Object **T_GDGR##**. The DataStore Object **T_GDGR##** has already been created in the BI system.

Assign the fields between your DataSource **ZGENEXT_##** and the DataStore Object **T_GDGR##** as follows:

Field Name	Enter
<i>SEQUENCENO</i>	0TCTREQSID
<i>CALDAY</i>	0CALDAY

Continued on next page

Field Name	Enter
MATERIAL	0MATERIAL
T_SLSQTY	0CRM_SALQTY
UNIT	0SALES_UNIT

Do not forget to change the aggregation type for Key Figure **0CRM_SALQTY** to **Overwrite**.

- a) In the context menu for the DataSource **ZGENEXT##**, choose *Create Transformation*.

Field Name	Enter
Object Type	DataStore Object
Name	T_GDGR##

Choose *Create Transformation* ✓

Now assign the fields between your DataSource **ZGENEXT##** and the DataStore Object **T_GDGR##** with connecting lines as follows:

Field Name	Enter
SEQUENCENO	0TCTREQSID
CALDAY	0CALDAY
MATERIAL	0MATERIAL
T_SLSQTY	0CRM_SALQTY
UNIT	0SALES_UNIT

- b) Change the rule of the Key Figure InfoObjet**0CRM_SALQTY** from *Summation* to *Overwrite* .

From the context menu of the InfoObject **0CRM_SALQTY**, choose *Rule Details* and in the *Aggregation* field use the dropdown box to change the aggregation type to *Overwrite*.

- c) Choose the button *Transfer Values* and you will be returned to the Transformation overview.
- d) Activate the Tranformation Rules, ignoring any warning messages that may appear, and go back to the initial screen by choosing the Activate icon.

Then click the *Previous Object* icon ↪ .

Continued on next page

3. Create an InfoPackage for DataSource ZGENEXT_## for use in loading data to the PSA. Use **ZGENEXT_## Delta Initialization** as the InfoPackage description. .

Set the Update Mode on the *Update* tab to **Initialize Delta Process – Initialization with Data Transfer**.

On *Schedule* tab start the data load immediately. Check your data load in the *Data Load Monitor*.

- a) Locate your DataSource **ZGENEXT_##**. From the context menu of your DataSource choose *Create InfoPackage* and in the presented dialog enter the name **ZGENEXT_## Delta Initialization**.
Choose Save  icon.
- b) In the context menu for the DataSource Object **T_GDGR##**, choose *Create Data Transfer Process*. All of the necessary information will be entered for you. You may change the name of the Data Transfer Process if you want to, but it is not necessary.
- c) On the *Processing* tab you will see that the option **Only PSA** is not available for change. In BI, InfoPackages only extract data to the PSA table.
- d) On the *Update* tab set the Update Mode to **Initialize Delta Process – Initialization with Data Transfer**.
- e) Ordinarily, the InfoPackage would not be scheduled on the *Schedule* tab, but the InfoPackage would be included in a Process Chain. In order to save time, in this exercise, we will not utilize a Process Chain, so on the *Schedule* tab, make sure the option **Start Data Load Immediately** is selected and then click the button Start.
- f) Within InfoPackage Maintenance (Scheduler), choose the *Monitor*  icon.
- g) On the tab page, choose *Detail* and the pushbutton *Refresh Details*  until the load process is finished.

Continued on next page

4. To load the data from the PSA table of DataSource **ZGENEXT_##** into the associated DataStore Object **T_GDGR##**, you have to create a Data Transfer Process. On the *Extraction* tab ensure that the *Extraction Mode* is set to **Delta**. Execute the Data Transfer Process.
 - a) Expand the tree under your DataSource **ZGENEXT_##** until you locate the DataStore Object **T_GDGR##**.
 - b) In the context menu for the DataStore Object **T_GDGR##**, choose *Create Data Transfer Process*. All of the necessary information will be entered for you. You may change the name of the Data Transfer Process if you want to, but it is not necessary.
 - c) In the Data Transfer Process, go to the *Extraction* tab and choose the *Delta* option from the dropdown box if it is not already displayed. Choose the *Activate*  icon to activate your Data Transfer Process.

 **Note:** You do not need to initialize the delta process explicitly for the delta transfer.
 - d) In the Data Transfer Process, go to the *Execute* tab and choose the *Execute*  icon.

 **Note:** If you do not include the Data Transfer Process in a Process Chain, you can execute it on the Extraction tab.
5. Check the status of the extraction and the contents of the DataStore Object in the *DTP Monitor* respectively within *DataStore Object Administration*.
 - a) From the Data Transfer Process **ZGENEXT_## / FB4CLNT800 → T_GDGR##**, choose the *Monitor*  icon to check the status of the extraction in the *DTP Monitor*. If necessary, choose the button *Refresh Details*  until the upload is finished. All lights should be green. Go back to the Data Warehousing Workbench using the *Back*  button.
 - b) From the context menu of your DataStore Object **T_GDGR##** choose *Manage*.
 - c) On the *Requests* tab, choose the *Activate*  button.
 - d) Select the RequestID by clicking the empty square to the left of the Request ID and choose the *Start*  icon. Close the activation dialog.
 - e) Go to the *Contents* tab and check the contents of the *Active Data* table by using the *Active Data* button. Then use the *Execute*  icon. The table should contain three records. Then choose the *Back*  icon two times.
6. The trainer starts the program **ZT_FILL_TABLE_GEN_EXTRACTION** with the parameter **DELTA** to generate new delta records in the table.

Continued on next page



Caution: Since there is only one program, participants should wait until the trainer has performed this step before proceeding.

- a) The trainer **only** will use transaction SE38 to execute the ABAP program.
7. Check the content of the table **ZTR_GENERIC_EXT** after the program has been executed with transaction SE16.
 - a) Check the new content of the table **ZTR_GENERIC_EXT** with transaction SE16:
Enter the transaction code **/NSE16**.
Choose the *Table Content* icon
8. Check the delta queue for the DataSource **ZGENEXT_##** and the status of the generic delta using transaction code RSA7.
 - a) Transaction: **/NRSA7**
Find your DataSource **ZGENEXT_##** and select it.
Select the button for your DataSource **ZGENEXT_##**.
Choose the *Exit* icon and go back to the Data Warehousing Workbench using transaction code **/NRSA1**
9. Create a second InfoPackage for your DataSource **ZGENEXT_##** in order to request the delta by loading the new contents of the table **ZTR_GENERIC_EXT** into the PSA table of your DataSource. Use

Continued on next page

ZGENEXT_## Delta as InfoPackage description. Set the Update Mode on the *Update* tab to **Delta Update**. On *Schedule* tab start the data load immediately. Check your data load in the *Data Load Monitor*.

- a) Locate your DataSource **ZGENEXT_##**. From the context menu of your DataSource choose *Create InfoPackage* and in the presented dialog enter the name **ZGENEXT_## Delta**. Choose  icon.
 - b) On the *Processing* tab you will see that the option **Only PSA** is not available for change. In BI, InfoPackages only extract data to the PSA table.
 - c) On the *Update* tab set the Update Mode to **Delta Update**.
 - d) Ordinarily, the InfoPackage would not be scheduled on the *Schedule* tab, but the InfoPackage would be included in a Process Chain. In order to save time, in this exercise, we will not utilize a Process Chain, so on the *Schedule* tab, make sure the option **Start Data Load Immediately** is selected and then click the button *Start*.
 - e) Within InfoPackage Maintenance (Scheduler), choose the *Monitor*  icon.
 - f) On the tab page, choose *Detail* and the pushbutton *Refresh Details*  until the load process is finished.
10. Load the delta from the PSA table into your DataStore Object **T_GDGR##** by executing the existing Data Transfer Process again.
- a) Expand the tree under your DataSource **ZGENEXT_##** until to you locate the DataStore Object **T_GDGR##**.
 - b) Find the Data Transfer Process **ZGENEXT_## / FB4CLNT800 → T_GDGR##** and go to the *Execute* tab and choose the *Execute*  icon.

Continued on next page

11. Check the status of the extraction and the contents of the DataStore Object in the *DTP Monitor* respectively within *DataStore Object Administration*.

- a) From the Data Transfer Process **ZGENEXT_## / FB4CLNT800 -> T_GDGR##**, choose the *Monitor*  icon to check the status of the extraction in the *DTP Monitor*.

If necessary, choose the button *Refresh Details*  until the upload is finished. All lights should be green. Go back to the Data Warehousing Workbench using the *Back*  button.

- b) From the context menu of your DataStore Object **T_GDGR##** choose *Manage*
- c) On the *Requests* tab, choose the *Activate*  button.
- d) Select the RequestID by clicking the empty square to the left of the Request ID and choose the *Start*  icon. Close the activation dialog.
- e) Go to the *Contents* tab and check the contents of the *Active Data* table by using the *Active Data* button. Then use the *Execute*  icon. The table should now contain seven records (three records from the delta init load and four delta records added by the delta load).

Then choose the *Back*  icon two times.



Lesson Summary

You should now be able to:

- Provide examples of how delta management works

Lesson: Additional Features of Delta Data Acquisition

Lesson Overview

This lesson explains additional aspects of delta management.



Lesson Objectives

After completing this lesson, you will be able to:

- Explain additional aspects of delta management and how they work

Business Example

Your company has recognized how useful it can be to integrate people, information and business processes in a heterogeneous system landscape and would like to benefit from this. Practice has shown, though, that loading very large datasets makes considerable demands on hardware and system performance.

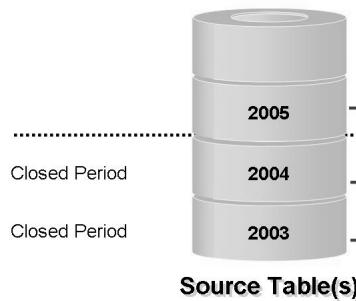
It is therefore necessary to examine if and how the data records can be loaded into BI with a delta process. First you must understand the modes of operation and the different variants of a delta loading process. You also need to clarify how the approach could look if classical delta management is not possible.

Additional Features of Delta Data Acquisition



Source System

Changes to data records are only possible/permitted > 2004



BI

Deletion of the current year
prior to loading the current
year will ensure data accuracy

Full Update:
Selection '2005 -999'
(with previous automatic
deletion of similar requests)

Full Update: Selection '2004'

Full Update: Selection '2003'

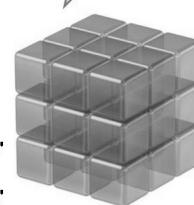


Figure 122: Delta Using Full Update: Snapshots

If a DataSource does not support a delta process, a **snapshot scenario** is possible:

In the delta (selection in full update mode), all data that might have changed is always loaded, for example, all the data for a period that is still open for posting.

With transaction data, the data that has been loaded so far for this period has to be deleted, otherwise the update will be carried out repeatedly.

For this scenario, the last data request can be deleted in BI automatically. Scheduling this loading process at regular intervals is also supported. The selection can be made dynamic by using variables in the InfoPackage (for example, you could make every selection with the current date using the change date in a master data table).

Additionally, the system can use the sigma icon on the *Data Targets* tab of the Info Package to determine whether or not it should automatically delete certain requests that match the new load.

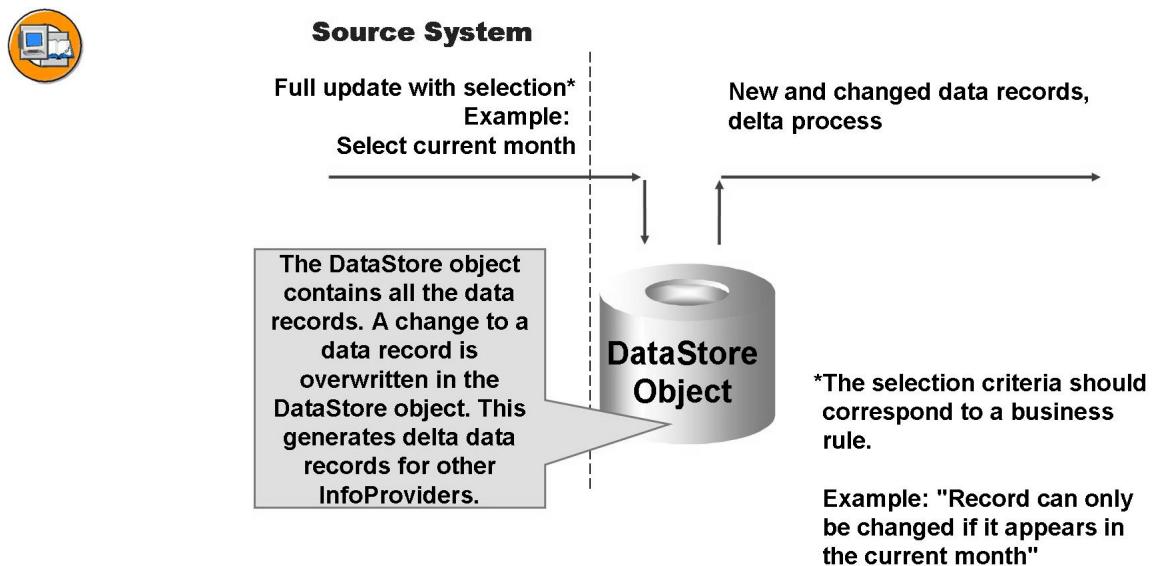


Figure 123: Delta Using Full Update: DataStore Object

If a DataSource does not support a delta process, you can generate delta data records using a DataStore object:

- In the delta (selection in full update mode), all data that might have changed is always loaded, for example, all the data for a period that is still open for posting. In the perfect scenario, the selection in the InfoPackage can refer directly to the changed data (for example, with selections using the change date of the data records, if this exists).
- The key figures are updated to the DataStore object using the “overwrite” update type in the update rules.
- If data records have been deleted, you may also have to delete these data records from the DataStore object too (by generating cancel records yourself).



Lesson Summary

You should now be able to:

- Explain additional aspects of delta management and how they work



Unit Summary

You should now be able to:

- Describe important elements of delta management for data extraction to BI
- Explain the role of the update mode and the properties of delta processes
- Make decisions on modeling a data flow dependent on the delta properties of the DataSource to be connected
- Manage the delta process effectively
- Explain the basic concepts of delta management and describe how it works
- Provide examples of how delta management works
- Explain additional aspects of delta management and how they work

Unit 5

Transfer of Flat Files

Unit Overview

In this unit you will see the options, functions and necessities for extraction of data from flat files.

The extraction of flat files is the most common method of integrating data from external systems into an BI system. Thus it is very important to get to know its functioning and special considerations.



Unit Objectives

After completing this unit, you will be able to:

- Explain the basic prerequisites of extraction using the file interface.
- Create your own file system
- Use enhanced functions for preparing, testing and loading files using the file adapter
- Decide, based on your knowledge of the source data, which process should be used during data loading
- Use a DataStore object to run delta load processes for the file interface

Unit Contents

Lesson: Transfer of Flat Files	304
Exercise 10: Extraction with Delta Processes	317

Lesson: Transfer of Flat Files

Lesson Overview

This lesson introduces flat file extraction and explains the enhanced functions for extraction using the file interface. Delta management options represent an important thematic focus.



Lesson Objectives

After completing this lesson, you will be able to:

- Explain the basic prerequisites of extraction using the file interface.
- Create your own file system
- Use enhanced functions for preparing, testing and loading files using the file adapter
- Decide, based on your knowledge of the source data, which process should be used during data loading
- Use a DataStore object to run delta load processes for the file interface

Business Example

You want to load data from an external system into BI using flat files. You also want to consider the option of loading this data by delta upload.

You also consider an alternative method. You can use the DB Connect functions for direct data extraction into BI from tables and views of a database management system that is directly connected to BI.

Data Acquisition from Flat Files

The figure shows the architecture for data acquisition.

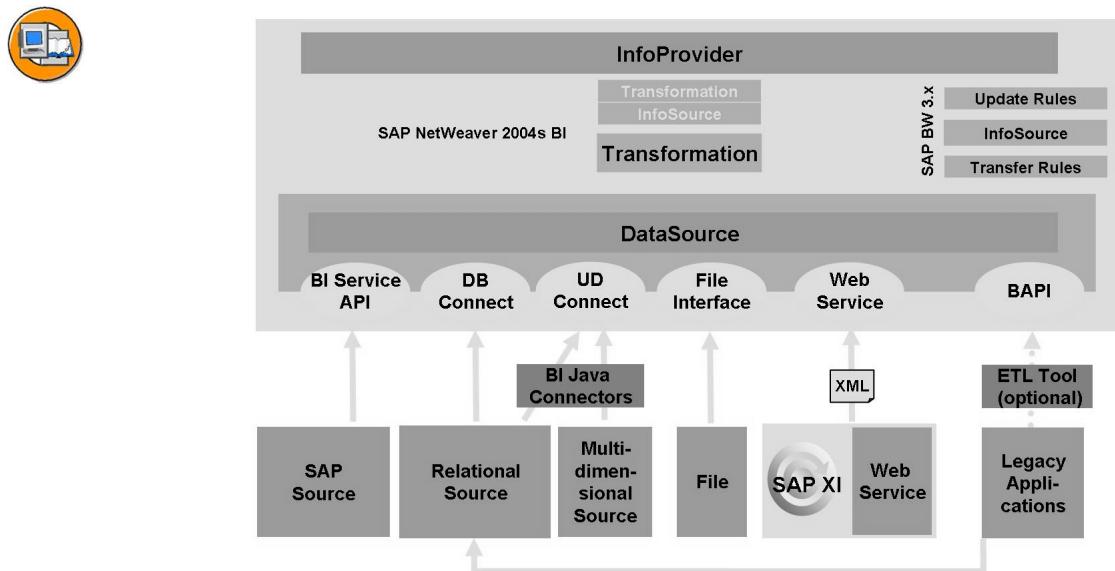


Figure 124: Data staging - architecture

Here you get an overview of the possible loading scenarios in BI. An important part of this is the flat-file interface.

All types of data can be loaded into BI with files. This includes text, master data attributes, hierarchies and transaction data.

BI supports the transfer of data from flat files, files in ASCII format (American Standard Code for Information Interchange) or CSV format (Comma Separated Value). For example, if budget planning for a company's branch offices is done in Microsoft Excel, this planning data can be loaded into BI so that a plan-actual comparison can be performed. The data for the flat file can be transferred to BI from a workstation or from an application server.

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 and with a zero (0) if they are numerical fields.
- If separators are used inconsistently in a CSV file, the wrong (not defined in the DataSource) separator is read as a character, and both fields are merged into one field and possibly shortened. Subsequent fields are then no longer in the correct order.

Note the following with regard to CSV files and ASCII files:

- The conversion routine being used determines whether you have to specify leading zeros.
- For dates, you usually use the format YYYYMMDD, without internal separators. Depending on the conversion routine being used, you can also use other formats.

Transaction data has no fixed format. The transfer structure in BI simply has to correspond with the file layout.

Hierarchy data and text data have fixed format requirements for the file, whereas with master data attributes the attribute data fields can exist in any sequence, providing they correspond with the fields in the file. The keys, however, must be at the beginning of the file layout.

Before you can load flat files, a file system must be created as the source system of the BI system. You can create a file system on the *Source Systems* tab in the Data Warehousing Workbench. You need to have only one “file” type source system. The actual location of the physical file is determined in the DataSource and can reside on the application server or in a file in the LAN/WAN.

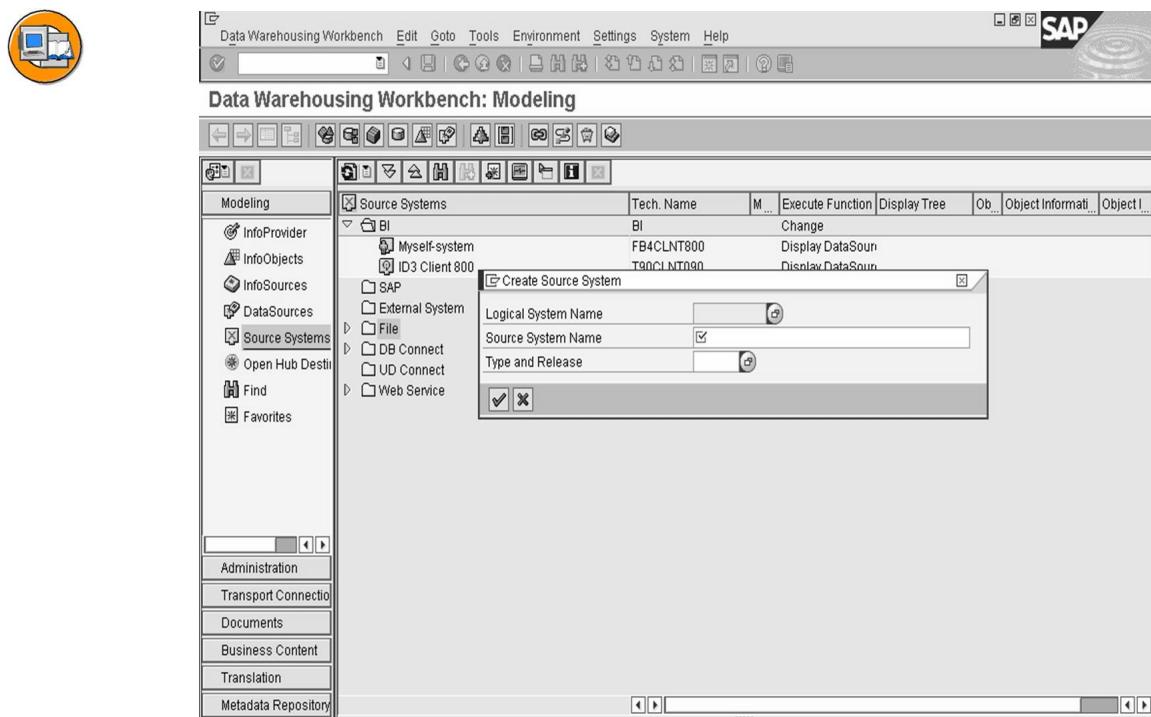


Figure 125: Creation of Flat File Source System

In addition to the functions already described, there are additional functions that ensure stable data extraction:

- Exclude certain (header) rows
- Selection options to filter the file when loading
- Routine for naming the physical file
- Extensive preview functions

File loading processes are extremely flexible. Not all of the functions named above are needed for every loading process. If they are needed, they are very helpful, however.

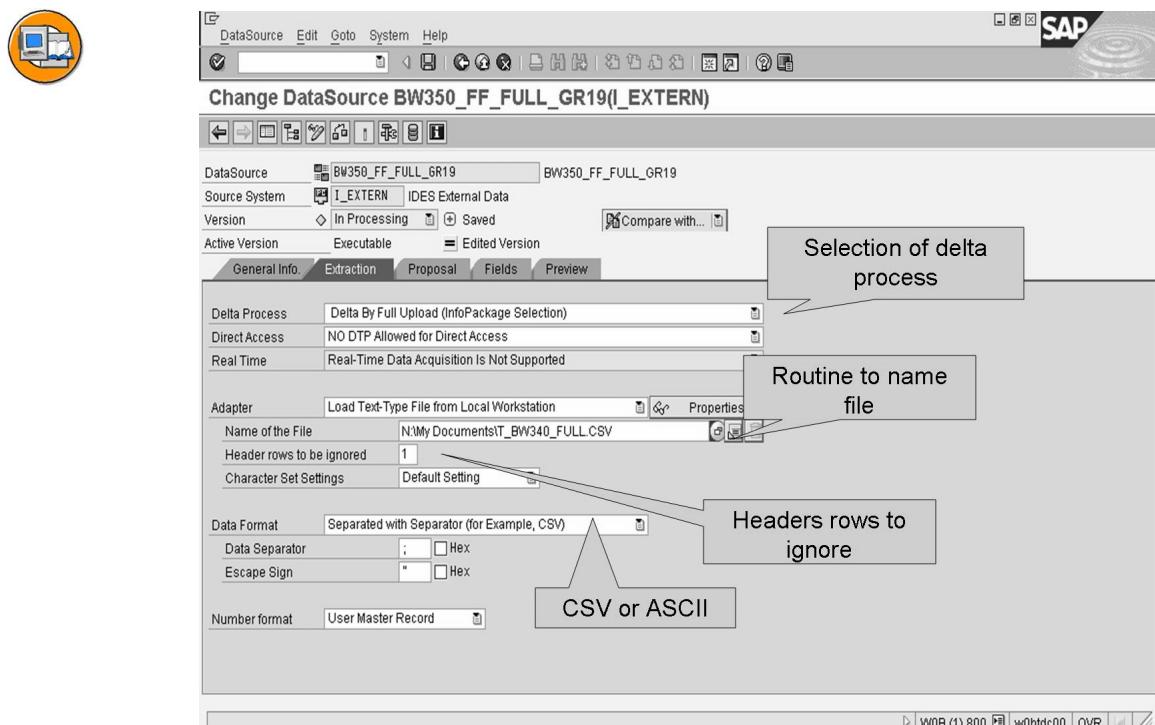


Figure 126: Flat File DataSource Loading Functions I

If you use the file DataSource, you need a process to create or regenerate the file needed by BI at the right time. In some instances this can lead to errors. For example, it can happen that the external application which provides the *sales data.csv* file contains the data from the previous month. This is a considerable problem because BI loads the wrong data and this might not even be noticed.

The *Extraction* tab contains the *Create Routine* pushbutton. When you use this pushbutton, a different file name can be generated every time with the help of an ABAP routine. Instead of *sales data.csv* file name, the routine would generate the name *JAN2005_sales data.csv*. The external application would then have to create the file with this dynamically generated name. If this is not the case, BI

cannot find the file and a loading error occurs. The corresponding e-mail warning messages are also displayed. The problem is detected in this way, and duplicated data is not loaded by mistake.



Hint: When you upload external data, you are able to load the data from any workstation into BI. For performance reasons, however, you should store the data on an application server and load it from there into BI. This means that you can also load the data in the background.

If you want to upload a large amount of transaction data from a flat file, and you are able to specify the file type of the flat file, you should create the flat file as an ASCII file. From a performance point of view, uploading the data from an ASCII file is the most cost-effective method. Under certain circumstances, generating an ASCII file might involve a larger workload

There is the additional option of specifying a logical file name. The physical file path and name can then be defined in the transaction FILE. Variables to determine the physical file name or path could be used there too. This is not a transaction that is specific to BI.

Without descriptions in the file it is not easy to check whether a file has the correct information, or even if the correct file was selected. The descriptions are normally found at the beginning of the file. They can contain column headers and details of the time of creation or the creator of the file. This **metadata** (descriptive information or data about data) should of course be ignored. You can control this using the *Number of Headers to Be Ignored* option on the *Extraction* tab of the flat file DataSource.

Data can be checked with a preview function. By clicking on the corresponding button you can see how the data looks when loading it in the preview, without actually loading the data with a InfoPackage. This is a very useful function that enables you to ensure that the technical field definitions are loaded in the correct sequence, without actually loading the data using the BI staging process.



Change DataSource URTEST(PM_FL_XX)

DataSource	URTEST	File DS																																																																													
Source System	PM_FL_XX	PM File Sources Group XX																																																																													
Version	new	Not Saved																																																																													
Active Version	Does Not Exist																																																																														
General Info Extraction Proposal Fields Preview																																																																															
Converter	Separated with Separator (for Example, CSV)																																																																														
No. of Data Records	3	Load Example Data																																																																													
File Output																																																																															
S... Data 1 CARRID;CONNID;FLDATE;BOOKID;CUSTOMID;CUSTTYPE;SMOKER;LUGGWEIGHT;WUNIT;INVOICE;CLASS;FORCURAM;FORCURKEY;LOCUR 2 LH;400;28.02.1995;1;1;B;;20;KG>C;899;DEM;899;DEM;26.02.1995 3 LH;400;28.02.1995;2;2;P;;20;KG>F;890;DEM;890;DEM;27.02.1995																																																																															
Field Proposal <table border="1"> <thead> <tr> <th>Co</th> <th>P.</th> <th>Field</th> <th>Descript.</th> <th>Data type</th> <th>Length</th> <th>Decim.</th> <th>Output...</th> <th>Conver.</th> <th>Format</th> <th>Unit</th> </tr> </thead> <tbody> <tr> <td><input checked="" type="checkbox"/></td> <td>1</td> <td>CARRID</td> <td>CARRID</td> <td>UNIT</td> <td>3</td> <td></td> <td></td> <td></td> <td>Internal</td> <td></td> </tr> <tr> <td><input checked="" type="checkbox"/></td> <td>2</td> <td>CONNID</td> <td>CONNID</td> <td>INT2</td> <td>6</td> <td></td> <td></td> <td></td> <td>Internal</td> <td></td> </tr> <tr> <td><input checked="" type="checkbox"/></td> <td>3</td> <td>FLDATE</td> <td>FLDATE</td> <td>DATS</td> <td>10</td> <td></td> <td>RSDAT</td> <td>External</td> <td></td> <td></td> </tr> <tr> <td><input checked="" type="checkbox"/></td> <td>4</td> <td>BOOKID</td> <td>BOOKID</td> <td>INT1</td> <td>3</td> <td></td> <td></td> <td></td> <td>Internal</td> <td></td> </tr> <tr> <td><input checked="" type="checkbox"/></td> <td>5</td> <td>CUSTOMID</td> <td>CUSTOMID</td> <td>INT1</td> <td>3</td> <td></td> <td></td> <td></td> <td>Internal</td> <td></td> </tr> <tr> <td><input checked="" type="checkbox"/></td> <td>6</td> <td>CUSTTYPE</td> <td>CUSTTYPE</td> <td>CHAR</td> <td>1</td> <td></td> <td></td> <td></td> <td>Internal</td> <td></td> </tr> </tbody> </table>			Co	P.	Field	Descript.	Data type	Length	Decim.	Output...	Conver.	Format	Unit	<input checked="" type="checkbox"/>	1	CARRID	CARRID	UNIT	3				Internal		<input checked="" type="checkbox"/>	2	CONNID	CONNID	INT2	6				Internal		<input checked="" type="checkbox"/>	3	FLDATE	FLDATE	DATS	10		RSDAT	External			<input checked="" type="checkbox"/>	4	BOOKID	BOOKID	INT1	3				Internal		<input checked="" type="checkbox"/>	5	CUSTOMID	CUSTOMID	INT1	3				Internal		<input checked="" type="checkbox"/>	6	CUSTTYPE	CUSTTYPE	CHAR	1				Internal	
Co	P.	Field	Descript.	Data type	Length	Decim.	Output...	Conver.	Format	Unit																																																																					
<input checked="" type="checkbox"/>	1	CARRID	CARRID	UNIT	3				Internal																																																																						
<input checked="" type="checkbox"/>	2	CONNID	CONNID	INT2	6				Internal																																																																						
<input checked="" type="checkbox"/>	3	FLDATE	FLDATE	DATS	10		RSDAT	External																																																																							
<input checked="" type="checkbox"/>	4	BOOKID	BOOKID	INT1	3				Internal																																																																						
<input checked="" type="checkbox"/>	5	CUSTOMID	CUSTOMID	INT1	3				Internal																																																																						
<input checked="" type="checkbox"/>	6	CUSTTYPE	CUSTTYPE	CHAR	1				Internal																																																																						

Figure 127: File Adapter Designtime Proposals

As there is no replication of metadata for a DataSource when you upload flat files, you must reproduce the field sequence in the DataSource.

You can specify how many data records you want to load and choose the *Load Sample Data* button. The data is displayed in the format of your file in the upper part of the tab page. The system displays the proposal for the field list in the lower part of the tab page. Where necessary, remove the *Copy to Field Lists* selection for fields that are not to be transferred to the field lists.

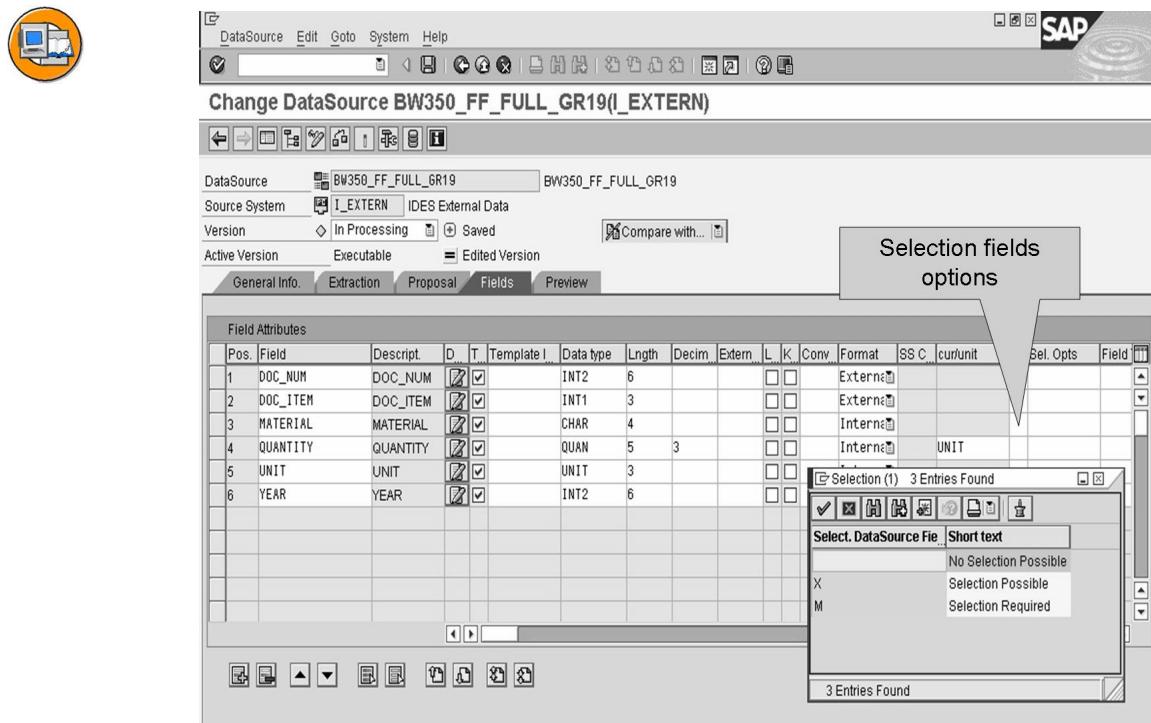


Figure 128: Selection Options for Filtering a File System

A DataSource for a SAP source system can be configured in such a way that the data is filtered before staging in BI and takes place in the source system. With file systems this is done by selecting the corresponding values from the selection list for the desired filter fields on the *Fields* tab of each DataSource.

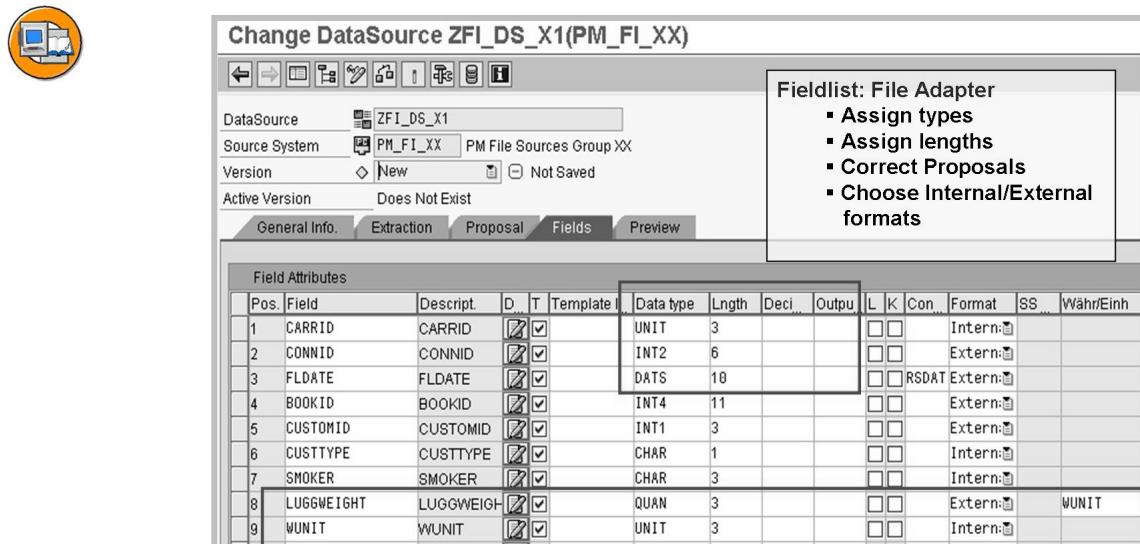


Figure 129: File Adapter Designtime Fieldlist

Options for handling field proposals:

- By setting the Transfer indicator, select the fields that should be provided for extraction.
- If required, specify InfoObjects of the BI system for the fields under Template InfoObject. The system proposes them for assignment to a field when defining the transformation. It is possible to transfer the technical properties of the InfoObject into the DataSource field.
- If required, change the data type of the field.
- If required, change the output length for a field.
- If required, change the values for the key fields of the source.
- Define if lowercase is supported.
- Specify whether the source provides the data in an internal or external format.
- If required, specify a conversion routine that converts the data to internal format.
- Select the fields for which you want to be able to set selection criteria when you schedule a data request using a BI InfoPackage. Data for this type of field is transferred in accordance with the selection criteria specified in the InfoPackage.
- Choose the selection options (such as EQ, BT) to be provided for selection in the InfoPackage. If required, in the field type, define if the data to be selected is language-dependent or time-dependent.

On the *Preview* tab, you can select *Read Preview Data* and data records, according to your field selection, is displayed in a preview. With this function you are able to check whether the data formats and data are correct.

CARRID	CONNID	FLDATE	BOOKID	CUSTOMID	CUSTTYPE	SMOKER	LUGGWEIGHT	WUNIT	INVOICE	CLASS	FORCURAM	FORCURKEY
LH	400	28.02.1995	1	1	B		20	KG	X	C	899,00	DEM
LH	400	28.02.1995	2	2	P		20	KG	X	F	890,00	DEM
LH	400	28.02.1995	3	3	P	X	30	KG	X	Y	850,00	DEM
LH	454	17.11.1995	1	3	P		20	KG	X	Y	1.450,00	DEM
LH	454	17.11.1995	2	1	B		20	KG	X	C	1.499,00	DEM
LH	455	06.06.1995	1	1	B		20	KG	X	C	1.090,00	USD

Figure 130: File Adapter Preview Example

You can make settings in BI so that the system expects and checks two different delta processes.

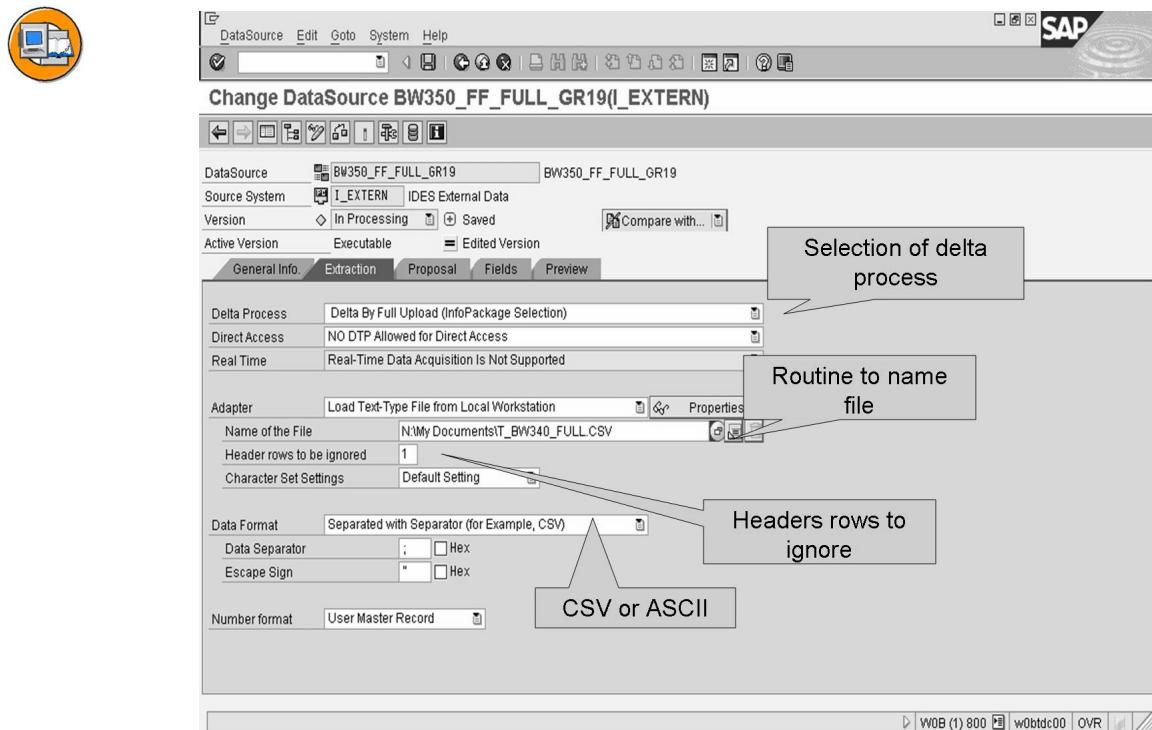


Figure 131: Flat File DataSource Loading Functions II

The delta process **Delta** with status “New” found on the *Extraction* tab sends the after image or new image of a record.

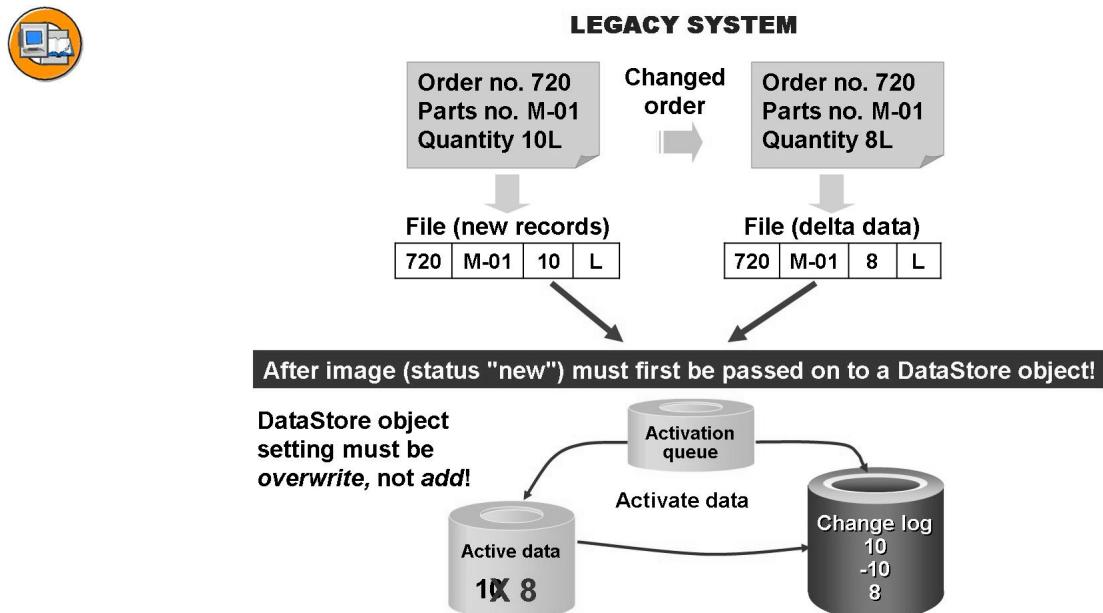


Figure 132: Delta Only with Status "New"

If a legacy system only recognizes the current (new) status of a change you have to use the DataStore object overwrite function.

After the data has been updated to the change log for the DataStore object (by activating the DataStore object data), it can be made available to other Standard InfoCubes or DataStore objects with the additive data which can then be processed by the respective target InfoProvider.

With the delta process **additive delta**, the BI system expects either a record with the difference or two records; one with the after image and one with the before image. The file system does not process the record mode, therefore the record mode X that would reverse the plus/minus sign cannot be used if the system is a SAP source system. The before image must show a reversed plus/minus sign for the key figures already.

If the DataSource is defined as additive, you have to use either an additive image or both images (before and after). In this case, a warning is displayed if the DataStore object is set to “overwrite”; you will be informed that this activity is not permitted. You can only use the “addition” setting in the DataStore object for additive file DataSources.

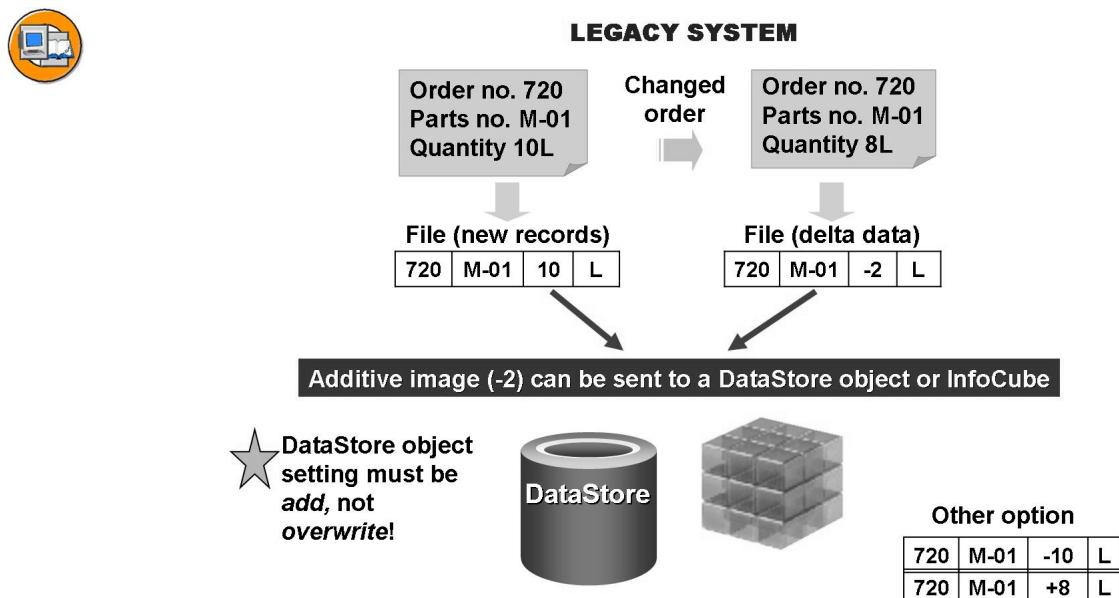


Figure 133: Additive Delta

If a DataSource is set to only perform a full load, this leads to an empty delta process. If the delta process is empty, the normal delta loading process is not available. There is a solution to this, however. You can work with a **pseudo delta**.

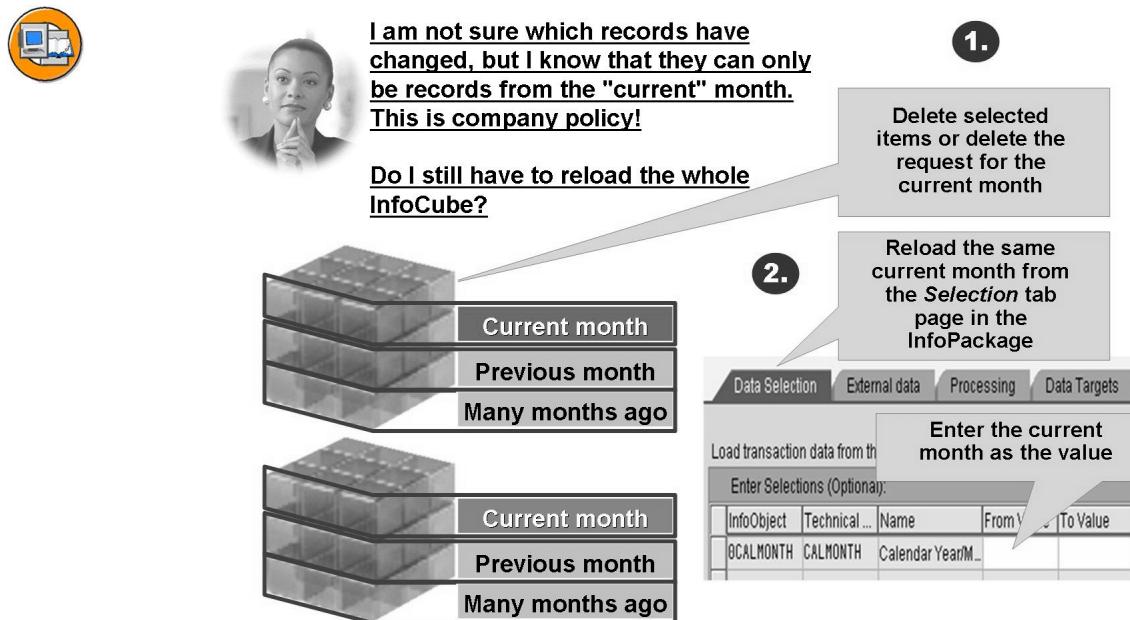


Figure 134: Pseudo Delta

A **pseudo delta** is not really a delta process at all, hence the adjective “pseudo”. The aim is not to constantly have to delete and reload the entire Standard InfoCube, but only the data that might have changed. The difficulty is in finding out which records might have changed. Internal or external (legal) regulations could demand that application data can no longer be changed after a particular period of time has elapsed. An example of this is accountancy data, which cannot normally be changed retrospectively, except perhaps for the current period.

Filters in the InfoPackage and the deletion of the same record area in the Standard InfoCube ensure that only changed data is reloaded, and that this data is not duplicated.

The processes that are required for a pseudo delta are time-consuming and prone to errors. You can automate the process to guarantee accuracy and consistency. In the first step, a variable is automatically used for the period to change the filter settings dynamically. In the example above, the SAP variable 0CMONTH is used to change the selected month automatically to the current month. You can also determine the selection criteria with a user-defined routine.

If the selection process was automated, you can use the function to automatically delete the same or similar requests to check whether requests that fulfill the specified criteria were already loaded. If a new request corresponds to an existing request, the existing request is deleted. In our example, this means: If you had already loaded the data for the current month, the data for the current month that was loaded beforehand would be deleted. This ensures that there is no duplicated data.

The criteria used to determine whether a request is a corresponding request can be limited to requests for full loads or initialization loads, to the DataSource with the same name, and so on. To do this, click on *Auto...* (on the *Data Targets* tab in the InfoPackage). (In most cases, you can only discern the word “Auto...”)



Note: This automation feature is only available for Standard InfoCubes. DataStore objects can filter out duplicate records resulting from loading the same or overlapping records again.

Exercise 10: Extraction with Delta Processes

Exercise Objectives

After completing this exercise, you will be able to:

- Display and simulate the loading process for a flat file to the DataStore Object in the preview mode
- Create a DataStore Object and fill it using a flat file DataSource with the help of different delta processes

Business Example

The extraction of a flat file can stage delta information in two ways. In this exercise you will first use a flat file that sends delta data with an additive image to the DataStore Object and second you will use a flat file that sends delta data with an after image to the DataStore Object. In both scenarios, you are working with a file that contains two orders for three different materials. The first order is in the year 2000, the second in the year 2001.

First, you begin with the file with an order for ten pieces. Then you use a delta load file that adds five additional pieces to each order and material.

Second, you will start again with the file with an order for ten pieces. Then you use a delta load file that overwrites the existing 10 000 pieces with 5.000 pieces for each order and material.

You also simulate the loading process for the file before you actually load the data.

Task 1:

Download the flat files which will provide the data for the following tasks.

1. Locate the files in the Shared Folders of the BI Business Workbench.

Use transaction SO04 and then the menu *Shared Folders → TRAINING: Material for BW Training → BW350: BW350 Files for BW Training*

Double click the folder *BW350: BW350 Files for BW Training* and then double click on the message *BW350 Files for Training* in the right hand pane.

Go to the *Attachments* tab and download the files by choosing *Export attachment* for each one.

Continued on next page

Task 2:

Create a DataStore Object as a copy.

1. Create the DataStore Object and give it the name and description **B350FF##**. Copy the DataStore Object **T_350FFO** and store the new DataStore Object in your InfoArea **T_BW350_GR##**.

Task 3:

Create a transaction data DataSource for the flat file source system **I_EXTERN** with delta process **FIL1 Delta via File Import with Delta Images**.

1. Create a DataSource **BW350_FF_ADDGR##** for the flat file source system **I_EXTERN**.
2. Use **BW350_FF_ADDGR##** as short, medium and long description for the new DataSource.
3. Set up the extraction details. The DataSource is intended to upload data from the CSV-Files downloaded in task 1 with delta process **FIL1 Delta via File Import with Delta Images**.
4. Create a field proposal for field names and date types by using the Load Example Data function. Ensure that all proposed fields are marked as being copied to the DataSource's field list.
5. Adjust the system's field proposal by entering the following Template InfoObjects:

<i>DOC_NUM</i>	0DOC_NUM
<i>DOC_ITEM</i>	0DOC_ITEM
<i>MATERIAL</i>	0MATERIAL
<i>QUANTITY</i>	0QUANTITY
<i>UNIT</i>	0UNIT
<i>YEAR</i>	0CALYEAR



Note: The usage of a Template InfoObject allows you to transfer the technical properties of the InfoObject into the DataSource field.

6. Specify the field *CALYEAR* as selection field.
7. Use the *Preview* function to check whether the data of the flat file is uploaded correctly

Continued on next page

Task 4:

Create a Transformation between your Data Source **BW350_FF_ADDGR##** and your DataStore object **B350FF##**.

1. From the context menu of your DataSource **BW350_FF_ADDGR##** create a Transformation between your Data Source **BW350_FF_ADDGR##** and your DataStore Object **B350FF##**.



Note: The system will create a mapping proposal using the Template InfoObjects specified previously in the DataSource maintenance.

2. Ensure that the aggregation type of Key Figure **0QUANTITY** is *Summation* (corresponding to delta process *FIL1 Delta via File Import with Delta Images* specified previously in DataSource maintenance).

Task 5:

Create an InfoPackage for use in loading data initially to the PSA table of the DataSource.

1. Create an InfoPackage for DataSource **BW350_FF_ADDGR##** for use in loading data from CSV-File **T_BW350_FULL** to the PSA. Use **FIL1 - Delta Initialization GR##** as InfoPackage description.

Set the Update Mode on the *Update* tab to **Initialize Delta Process – Initialization with Data Transfer**.

On *Schedule* tab start the data load immediately. Check your data load in the Data Load Monitor.

Task 6:

Create a Data Transfer Process to load PSA data into your DataStore Object. Check the uploaded data in the DataStore Object.

1. To load the data from the PSA table of DataSource **BW350_FF_ADDGR##** into the associated DataStore Object **B350FF##**, you have to create a Data Transfer Process. On the *Extraction* tab ensure that the *Extraction Mode* is set to Delta. Execute the Data Transfer Process.
2. Check the status of the extraction and the contents of the DataStore Object in the *DTP Monitor* respectively within *DataStore Object Administration*.



Note: The uploaded data will not be activated automatically in the DataStore Object **B350FF##**.

Continued on next page

Task 7:

Create an InfoPackage for use in loading delta data to the PSA table of the DataSource.

1. Create a second InfoPackage for your DataSource **BW350_FF_ADDGR##** in order to request the delta by loading the second CSV-File **T_BW350_DELTA_LOAD** with delta data into the PSA table of your DataSource. Use **FIL1 - Delta GR##** as InfoPackage description.

Set the *Update Mode* on the *Update* tab to **Delta Update**. On *Schedule* tab start the data load immediately. Check your data load in the *Data Load Monitor*.

Task 8:

Load the delta from the PSA table into your DataStore Object **B350FF##** by executing the existing Data Transfer Process again. Check the uploaded data in the DataStore Object

1. Execute the Data Transfer Process **BW350_FF_ADDGR## / I_EXTERN -> B350FF##** again.
2. Check the status of the extraction and the contents of the DataStore Object in the *DTP Monitor* respectively within *DataStore Object Administration*.



Note: The uploaded data will not be activated automatically in the DataStore Object **B350FF##**.

Task 9:

Delete the data from DataStore Object **B350FF##**.

1. Delete the data from DataStore Object **B350FF##**.

Task 10:

Create a transaction data DataSource for the flat file source system **I_EXTERN** with delta process **FIL0 Delta via File Import with After Images**.

1. Create a DataSource **BW350_FF_NEWR##** for the flat file source system **I_EXTERN**.
2. Use **BW350_FF_NEWDGR##** as short, medium and long description for the new DataSource.
3. Set up the extraction details. The DataSource is intended to upload data from the CSV-Files downloaded in task 1 with delta process **FIL0 Delta via File Import with After Images**.

Continued on next page

4. Create a field proposal for field names and date types by using the Load Example Data function. Ensure that all proposed fields are marked as being copied to the DataSource's field list.
5. Adjust the system's field proposal by entering the following Template InfoObjects:

<i>DOC_NUM</i>	0DOC_NUM
<i>DOC_ITEM</i>	0DOC_ITEM
<i>MATERIAL</i>	0MATERIAL
<i>QUANTITY</i>	0QUANTITY
<i>UNIT</i>	0UNIT
<i>YEAR</i>	0CALYEAR



Note: The usage of a Template InfoObject allows you to transfer the technical properties of the InfoObject into the DataSource field.

6. Specify the field *CALYEAR* as selection field.
7. Use the *Preview* function to check whether the data of the flat file is uploaded correctly

Task 11:

Create a Transformation between your Data Source **BW350_FF_NEWGR##** and your DataStore object **B350FF##**.

1. From the context menu of your DataSource **BW350_FF_NEWGR##** create a Transformation between your Data Source **BW350_FF_NEWGR##** and your DataStore Object **B350FF##**.



Note: The system will create a mapping proposal using the Template InfoObjects specified previously in the DataSource maintenance.

2. Change the aggregation type of Key Figure **0QUANTITY** to *Overwrit* (corresponding to delta process *FIL0 Delta via File Import with After-Images* specified previously in DataSource maintenance).

Continued on next page

Task 12:

Create an InfoPackage for use in loading data initially to the PSA table of the DataSource. Restrict the data to be uploaded to CALYEAR = 2001.

1. Create an InfoPackage for DataSource **BW350_FF_NEWGR##** for use in loading data from CSV-File **T_BW350_FULL** to the PSA. Use **FIL0 - Delta Initialization GR##** as InfoPackage description.

Set the Update Mode on the *Update* tab to **Initialize Delta Process – Initialization with Data Transfer**.

On *Schedule* tab start the data load immediately. Check your data load in the Data Load Monitor.

Task 13:

Create a Data Transfer Process to load PSA data into your DataStore Object. Check the uploaded data in the DataStore Object.

1. To load the data from the PSA table of DataSource **BW350_FF_NEWGR##** into the associated DataStore Object **B350FF##**, you have to create a Data Transfer Process. On the *Extraction* tab ensure that the *Extraction Mode* is set to Delta. Execute the Data Transfer Process.
2. Check the status of the extraction and the contents of the DataStore Object in the *DTP Monitor* respectively within *DataStore Object Administration*.

 **Note:** The uploaded data will not be activated automatically in the DataStore Object **B350FF##**.

Task 14:

Create an InfoPackage for use in loading delta data to the PSA table of the DataSource.

1. . Create a second InfoPackage for your DataSource **BW350_FF_NEWGR##** in order to request the delta by loading the second CSV-File **T_BW350_DELTA_LOAD** with delta data into the PSA table of your DataSource. Use **FIL0 - Delta GR##** as InfoPackage description.

Set the *Update Mode* on the *Update* tab to **Delta Update**. On *Schedule* tab start the data load immediately. Check your data load in the *Data Load Monitor*.

Continued on next page

Task 15:

Load the delta from the PSA table into your DataStore Object **B350FF##** by executing the existing Data Transfer Process again. Check the uploaded data in the DataStore Object

1. Execute the Data Transfer Process **BW350_FF_NEWGR## / I_EXTERN -> B350FF##** again.
2. Check the status of the extraction and the contents of the DataStore Object in the *DTP Monitor* respectively within *DataStore Object Administration*.



Note: The uploaded data will not be activated automatically in the DataStore Object **B350FF##**.

Solution 10: Extraction with Delta Processes

Task 1:

Download the flat files which will provide the data for the following tasks.

1. Locate the files in the Shared Folders of the BI Business Workbench.

Use transaction SO04 and then the menu *Shared Folders → TRAINING: Material for BW Training → BW350: BW350 Files for BW Training*

Double click the folder *BW350: BW350 Files for BW Training* and then double click on the message *BW350 Files for Training* in the right hand pane.

Go to the *Attachments* tab and download the files by choosing *Export attachment* for each one.

- a) Use transaction SO04 and then the menu *Shared Folders → TRAINING: Material for BW Training → BW350: BW350 Files for BW Training*
- b) Double click the folder *BW350: BW350 Files for BW Training* and then double click on the message *BW350 Files for Training* in the right hand pane.
- c) Go to the *Attachments* tab and download the following files by choosing *Export attachment* for each one.

T_BW350_FULL

T_BW350_DELTA_LOAD

Keep in mind the file system path where the flat files are exported to.

File System Path: _____

Continued on next page

Task 2:

Create a DataStore Object as a copy.

1. Create the DataStore Object and give it the name and description **B350FF##**. Copy the DataStore Object **T_350FFO** and store the new DataStore Object in your InfoArea **T_BW350_GR##**.
 - a) From the menu, choose *Data Warehousing Workbench* → *Modeling* → *InfoProvider*.
 - b) Expand the InfoArea tree by choosing *InfoProvider* → *BW_TRAINING* → *BW_CUSTOMER TRAINING* → *T_BW350*.
 - c) Select your InfoArea **T_BW350_GR##**, right-click and choose *Create DataStore Object*.

Field Name	Values
<i>DataStore Object</i>	B350FF##
<i>Long Description</i>	B350FF##
<i>Copy From</i>	T_350FFO

Choose the *Create*  icon.

Check the settings and fields in the DataStore Object, but do not change anything.

Choose the *Activate*  icon.

Continued on next page

Task 3:

Create a transaction data DataSource for the flat file source system **I_EXTERN** with delta process **FIL1 Delta via File Import with Delta Images**.

1. Create a DataSource **BW350_FF_ADDGR##** for the flat file source system **I_EXTERN**.
 - a) Use the menu *Data Warehousing Workbench → Modeling → Source Systems* and open the folder *File* to find the source system **I_EXTERN**.
 - b) From the context menu of **I_EXTERN** choose the option *Display DataSource Tree* and in the resulting DataSource tree find your Application Component **ZT_BW350_GR##**.
 - c) From the context menu of your Application Component **ZT_BW350_GR##** choose the option *Create DataSource* and in the resulting dialog box enter the following values:

<i>DataSource</i>	BW350_FF_ADDGR##
<i>Source System</i>	I_EXTERN
<i>Data Type DataSource</i>	Transaction Data

Click the *Transfer (Enter)*  icon.

2. Use **BW350_FF_ADDGR##** as short, medium and long description for the new DataSource.
 - a) On the *General Info* tab, enter the following value from the table and leave the defaults:

<i>Short description</i>	BW350_FF_ADDGR##
<i>Medium description</i>	BW350_FF_ADDGR##
<i>Long description</i>	BW350_FF_ADDGR##

Continued on next page

3. Set up the extraction details. The DataSource is intended to upload data from the CSV-Files downloaded in task 1 with delta process **FIL1 Delta via File Import with Delta Images**.

- a) On the *Extraction* tab, enter the following value from the table and leave the defaults:

<i>Delta Process</i>	FIL1 Delta via File Import with Delta Images
<i>Name of the File</i>	<individual path>T_BW350_FULL.CSV
<i>Header rows to be ignored</i>	1
<i>Data Format</i>	Separated with Separator (for Example: CSV)
<i>Number format</i>	Direct Entry
<i>Thousands Separator</i>	,
<i>Decimal Point Separator</i>	.

4. Create a field proposal for field names and date types by using the Load Example Data function. Ensure that all proposed fields are marked as being copied to the DataSource's field list.

- a) On the *Proposal* tab, click the *Load Example Data* icon.



Note: In the *Field Proposal* area, BI will propose field names and date types from the header row respectively the loaded sample data of the flat file.

Ensure that all proposed fields are marked as being copied to the DataSource's field list in the most-left column of the *Field Proposal* area.

5. Adjust the system's field proposal by entering the following Template InfoObjects:

<i>DOC_NUM</i>	0DOC_NUM
<i>DOC_ITEM</i>	0DOC_ITEM
<i>MATERIAL</i>	0MATERIAL

Continued on next page

QUANTITY	0QUANTITY
UNIT	0UNIT
YEAR	0CALYEAR



Note: The usage of a Template InfoObject allows you to transfer the technical properties of the InfoObject into the DataSource field.

- a) On the *Fields* tab enter the Template InfoObjects as given above.
- b) When entering the Template InfoObjects you will be presented with a dialog box asking whether you want to copy the field properties from the InfoObject. Mark the check box *Do Not Show this Question Again in This Session* and choose the button *Copy*.
6. Specify the field *CALYEAR* as selection field.
 - a) On the *Fields* tab enter 'X' in column Selection for field *CALYEAR*.
7. Use the *Preview* function to check whether the data of the flat file is uploaded correctly
 - a) On the *Preview* tab, click the *Read Preview Data* icon. In the resulting dialog box, you will be asked to first activate the DataSource, by clicking the *Activate*  icon.

The BI system will then read the flat file data and you will see a preview.

Continued on next page

Task 4:

Create a Transformation between your Data Source **BW350_FF_ADDGR##** and your DataStore object **B350FF##**.

1. From the context menu of your DataSource **BW350_FF_ADDGR##** create a Transformation between your Data Source **BW350_FF_ADDGR##** and your DataStore Object **B350FF##**.



Note: The system will create a mapping proposal using the Template InfoObjects specified previously in the DataSource maintenance.

- a) Use menu *Data Warehousing Workbench: Modeling* → *DataSources* → *BW Training* → *BW350 BI Data Acquisition* → *Group##* to find your DataSource **BW350_FF_ADDGR##**.
- b) From the context menu of your DataSource choose *Create Transformation*.
- c) In the *Create Transformation* dialog box enter the following values:

Object Type	DataStore Object
Name	B350FF##

Accept all other defaults and then click the *Transfer (Enter)* icon.

- d) Ensure that the mapping between the DataSource fields and the InfoObjects of the DataStore Object is as follows:

Field	Template InfoObject
<i>DOC_NUM</i>	ODOC_NUM
<i>DOC_ITEM</i>	ODOC_ITEM
<i>MATERIAL</i>	OMATERIAL
<i>QUANTITY</i>	OQUANTITY
<i>UNIT</i>	OUNIT
<i>MTH</i>	OCALYEAR

Continued on next page

2. Ensure that the aggregation type of Key Figure **0QUANTITY** is *Summation* (corresponding to delta process *FIL1 Delta via File Import with Delta Images* specified previously in DataSource maintenance).
 - a) From the context menu of the InfoObject **0QUANTITY**, choose *Rule Details* and check whether the entry in the Aggregation field is *Summation*.
 - b) Activate the Transformation by choosing the *Activate*  icon.

Task 5:

Create an InfoPackage for use in loading data initially to the PSA table of the DataSource.

1. Create an InfoPackage for DataSource **BW350_FF_ADDGR##** for use in loading data from CSV-File **T_BW350_FULL** to the PSA. Use **FIL1 - Delta Initialization GR##** as InfoPackage description.

Set the Update Mode on the *Update* tab to **Initialize Delta Process – Initialization with Data Transfer**.

On *Schedule* tab start the data load immediately. Check your data load in the Data Load Monitor.

- a) Locate your DataSource **BW350_FF_ADDGR##**. From the context menu of your DataSource choose *Create InfoPackage* and in the presented dialog enter the name **FIL 1 - Delta Initialization GR##**. Choose  icon..
- b) On the *Processing* tab you will see that the option **Only PSA** is not available for change. In BI, InfoPackages only extract data to the PSA table..
- c) On the *Update* tab set the Update Mode **Initialize Delta Process – Initialization with Data Transfer**.
Then choose the *Continue (Enter)*  icon.
- d) Ordinarily, the InfoPackage would not be scheduled on the *Schedule* tab, but the InfoPackage would be included in a Process Chain. In order to save time, in this exercise, we will not utilize a Process Chain, so on the *Schedule* tab, make sure the option **Start Data Load Immediately** is selected and then click the button *Start*.
- e) Within InfoPackage Maintenance (Scheduler), choose the *Monitor*  icon.
- f) On the tab page, choose *Detail* and the pushbutton *Refresh Details*  until the load process is finished. All lights should be green.

Continued on next page

Task 6:

Create a Data Transfer Process to load PSA data into your DataStore Object. Check the uploaded data in the DataStore Object.

1. To load the data from the PSA table of DataSource **BW350_FF_ADDGR##** into the associated DataStore Object **B350FF##**, you have to create a Data Transfer Process. On the *Extraction* tab ensure that the *Extraction Mode* is set to Delta. Execute the Data Transfer Process.
 - a) Expand the tree under your DataSource **BW350_FF_ADDGR##** until you locate the DataStore Object **B350FF##**.
 - b) In the context menu for the DataStore Object **B350FF##**, choose *Create Data Transfer Process*. All of the necessary information will be entered for you. You may change the name of the Data Transfer Process if you want to, but it is not necessary.

Choose the *Continue (Enter)*  icon.
 - c) In the Data Transfer Process, go to the *Extraction* tab and choose the *Delta* option from the dropdown box if it is not already displayed. Choose the *Activate*  icon to activate your Data Transfer Process.

 **Note:** You do not need to initialize the delta process explicitly for the delta transfer.
 - d) In the Data Transfer Process, go to the *Execute* tab and choose the *Execute*  icon.

 **Note:** If you do not include the Data Transfer Process in a Process Chain, you can execute it on the *Extraction* tab.

Continued on next page

2. Check the status of the extraction and the contents of the DataStore Object in the *DTP Monitor* respectively within *DataStore Object Administration*.



Note: The uploaded data will not be activated automatically in the DataStore Object **B350FF##**.

- a) From the Data Transfer Process **BW350_FF_ADDGR## / I_EXTERN -> B350FF##**, choose the *Monitor* icon to check the status of the extraction in the *DTP Monitor*. If necessary, choose the button *Refresh Details* until the upload is finished. All lights should be green. Go back to the Data Warehousing Workbench using the *Back* button.
- b) From the context menu of your DataStore Object **B350FF##** choose *Manage* and go to the *Contents* tab and check the contents of the *Activation Queue (New Data)*, *Active Data and Change Log* tables in succession. The Activation Queue should have six records and the Active Data and Change Log tables should be empty.
- c) Go to the *Requests* tab and choose the *Activate* icon.
- d) Select the RequestID by clicking the empty square to the left of the RequestID and choose the *Start* icon.
- e) Go to the *Contentstab* and check the contents of the *Activation Queue (New Data)*, *Active Data and Change Log* tables in succession again. The Activation Queue should be empty and the Active Data and Change Log tables should each have six records.

Task 7:

Create an InfoPackage for use in loading delta data to the PSA table of the DataSource.

1. Create a second InfoPackage for your DataSource **BW350_FF_ADDGR##** in order to request the delta by loading the second CSV-File **T_BW350_DELTA_LOAD** with delta data into the PSA table of your DataSource. Use **FIL1 - Delta GR##** as InfoPackage description.

Continued on next page

Set the *Update Mode* on the *Update* tab to **Delta Update**. On *Schedule* tab start the data load immediately. Check your data load in the *Data Load Monitor*.

- a) Locate your DataSource **BW350_FF_ADDGR##**. From the context menu of your DataSource choose *Create InfoPackage* and in the presented dialog enter the name *FIL1 - Delta GR##*. Choose  icon.
- b) On the *Extraction* tab, enter the following value for *Name of File* and accept the defaults:

<i>Name of the File</i>	<Individual Path> T_BW350_DELTA_LOAD.CSV
-------------------------	---

- c) On the *Processing* tab you will see that the option **Only PSA** is not available for change. In BI, InfoPackages only extract data to the PSA table
- d) On the *Update* tab set the Update Mode **Delta Update**.
- e) Ordinarily, the InfoPackage would not be scheduled on the *Schedule* tab, but the InfoPackage would be included in a Process Chain. In order to save time, in this exercise, we will not utilize a Process Chain, so on the *Schedule* tab, make sure the option **Start Data Load Immediately** is selected and then click the button *Start*.
- f) Within InfoPackage Maintenance (Scheduler), choose the  icon.
- g) On the tab page, choose *Detail* and the pushbutton  until the load process is finished. All lights should be green.

Task 8:

Load the delta from the PSA table into your DataStore Object **B350FF##** by executing the existing Data Transfer Process again. Check the uploaded data in the DataStore Object

1. Execute the Data Transfer Process **BW350_FF_ADDGR## / I_EXTERN -> B350FF##** again.
 - a) Expand the tree under your DataSource **BW350_FF_ADDGR##** until you locate the DataStore Object **B350FF##**.
 - b) Find the Data Transfer Process **BW350_FF_ADDGR## / I_EXTERN -> B350FF##** and go to the Execute tab and choose the  icon.

Continued on next page

2. Check the status of the extraction and the contents of the DataStore Object in the *DTP Monitor* respectively within *DataStore Object Administration*.



Note: The uploaded data will not be activated automatically in the DataStore Object **B350FF##**.

- a) From the Data Transfer Process **BW350_FF_ADDGR## / I_EXTERN -> B350FF##**, choose the *Monitor* icon to check the status of the extraction in the *DTP Monitor*.
If necessary, choose the button *Refresh Details* until the upload is finished. All lights should be green.
- b) Go back to the Data Warehousing Workbench using the *Back* button.
- c) From the context menu of your DataStore Object **B350FF##** choose *Manage* and go to the *Contents* tab and check the contents of the *Activation Queue (New Data)*, *Active Data and Change Log* tables in succession. The Activation Queue should contain the six delta records (Quantity = 5.000) The Active Data and Change Log tables should be unchanged (Quantity = 10.000).
- d) Go to the *Requests* tab and choose the *Activate* button.
- e) Select the RequestID by clicking the empty square to the left of the Request ID and choose the *Start* icon.
- f) Go to the *Contents* tab and check the contents of the *Activation Queue (New Data)*, *Active Data and Change Log* tables in succession again. The Activation Queue should be empty The Active Data should contain six records (Quantity = 15.000). The Change Log should contain 18 records (6x New Image, 6x Before Image, 6x After Image).

Task 9:

Delete the data from DataStore Object **B350FF##**.

1. Delete the data from DataStore Object **B350FF##**.
 - a) Use menu *Data Warehousing Workbench: Modeling → InfoProvider → BW Training → BW Customer Training → BW350 BI Data Acquisition → BW350 Group## → Context menu of your DataStore Object B350FF## → Choose Delete Data*
 - b) In the *Delete* contents dialog box choose *Yes*.

Continued on next page

Task 10:

Create a transaction data DataSource for the flat file source system **I_EXTERN** with delta process **FIL0 Delta via File Import with After Images**.

1. Create a DataSource **BW350_FF_NEWGR##** for the flat file source system **I_EXTERN**.
 - a) Use the menu *Data Warehousing Workbench → Modeling → Source Systems* and open the folder *File* to find the source system **I_EXTERN**.
 - b) From the context menu of **I_EXTERN** choose the option *Display DataSource Tree* and in the resulting DataSource tree find your Application Component **ZT_BW350_GR##**.
 - c) From the context menu of your Application Component **ZT_BW350_GR##** choose the option *Create DataSource* and in the resulting dialog box enter the following values:

<i>DataSource</i>	BW350_FF_NEWGR##
<i>Source System</i>	I_EXTERN
<i>Data Type DataSource</i>	Transaction Data

Click the *Transfer (Enter)* icon.

2. Use **BW350_FF_NEWDGR##** as short, medium and long description for the new DataSource.
 - a) On the *General Info* tab, enter the following value from the table and leave the defaults:

<i>Short description</i>	BW350_FF_NEWGR##
<i>Medium description</i>	BW350_FF_NEWGR##
<i>Long description</i>	BW350_FF_NEWGR##

Continued on next page

3. Set up the extraction details. The DataSource is intended to upload data from the CSV-Files downloaded in task 1 with delta process **FIL0 Delta via File Import with After Images**.

- a) On the *Extraction* tab, enter the following value from the table and leave the defaults:

<i>Delta Process</i>	FIL0 Delta via File Import with After Images
<i>Name of the File</i>	<individual path>T_BW350_FULL.CSV
<i>Header rows to be ignored</i>	1
<i>Data Format</i>	Separated with Separator (for Example: CSV)
<i>Number format</i>	Direct Entry
<i>Thousands Separator</i>	,
<i>Decimal Point Separator</i>	.

4. Create a field proposal for field names and date types by using the Load Example Data function. Ensure that all proposed fields are marked as being copied to the DataSource's field list.

- a) On the *Proposal* tab, click the *Load Example Data* icon.



Note: In the *Field Proposal* area, BI will propose field names and date types from the header row respectively the loaded sample data of the flat file.

Ensure that all proposed fields are marked as being copied to the DataSource's field list in the most-left column of the *Field Proposal* area.

5. Adjust the system's field proposal by entering the following Template InfoObjects:

<i>DOC_NUM</i>	0DOC_NUM
<i>DOC_ITEM</i>	0DOC_ITEM
<i>MATERIAL</i>	0MATERIAL

Continued on next page

QUANTITY	OQUANTITY
UNIT	OUNIT
YEAR	OCALYEAR



Note: The usage of a Template InfoObject allows you to transfer the technical properties of the InfoObject into the DataSource field.

- a) On the *Fields* tab enter the Template InfoObjects as given above.
 - b) When entering the Template InfoObjects you will be presented with a dialog box asking whether you want to copy the field properties from the InfoObject. Mark the check box *Do Not Show this Question Again in This Session* and choose the button *Copy*.
6. Specify the field *CALYEAR* as selection field.
 - a) On the *Fields* tab enter 'X' in column Selection for field *CALYEAR*.
 7. Use the *Preview* function to check whether the data of the flat file is uploaded correctly
 - a) On the *Preview* tab, click the *Read Preview Data* icon. In the resulting dialog box, you will be asked to first activate the DataSource, by clicking the *Activate*  icon.

The BI system will then read the flat file data and you will see a preview.

Continued on next page

Task 11:

Create a Transformation between your Data Source **BW350_FF_NEWGR##** and your DataStore object **B350FF##**.

1. From the context menu of your DataSource **BW350_FF_NEWGR##** create a Transformation between your Data Source **BW350_FF_NEWGR##** and your DataStore Object **B350FF##**.



Note: The system will create a mapping proposal using the Template InfoObjects specified previously in the DataSource maintenance.

- a) Use menu *Data Warehousing Workbench: Modeling* → *DataSources* → *BW Training* → *BW350 BI Data Acquisition* → *Group##* to find your DataSource **BW350_FF_NEWGR##**.
- b) From the context menu of your DataSource choose *Create Transformation*.
- c) In the *Create Transformation* dialog box enter the following values:

Object Type	DataStore Object
Name	B350FF##

Accept all other defaults and then click the *Transfer (Enter)* icon.

- d) Ensure that the mapping between the DataSource fields and the InfoObjects of the DataStore Object is as follows:

Field	Template InfoObject
<i>DOC_NUM</i>	0DOC_NUM
<i>DOC_ITEM</i>	0DOC_ITEM
<i>MATERIAL</i>	0MATERIAL
<i>QUANTITY</i>	0QUANTITY
<i>UNIT</i>	0UNIT
<i>MTH</i>	0CALYEAR

Continued on next page

2. Change the aggregation type of Key Figure **0QUANTITY** to *Overwrite* (corresponding to delta process **FIL0 Delta via File Import with After-Images** specified previously in DataSource maintenance).
 - a) From the context menu of the InfoObject **0QUANTITY**, choose *Rule Details* and check whether the entry in the Aggregation field is *Overwrite*.
 - b) Activate the Transformation by choosing the *Activate*  icon.

Task 12:

Create an InfoPackage for use in loading data initially to the PSA table of the DataSource. Restrict the data to be uploaded to CALYEAR = 2001.

1. Create an InfoPackage for DataSource **BW350_FF_NEWGR##** for use in loading data from CSV-File **T_BW350_FULL** to the PSA. Use **FIL0 - Delta Initialization GR##** as InfoPackage description.

Set the Update Mode on the *Update* tab to **Initialize Delta Process – Initialization with Data Transfer**.

Continued on next page

On *Schedule* tab start the data load immediately. Check your data load in the Data Load Monitor.

- a) Locate your DataSource **BW350_FF_NEWGR##**. From the context menu of your DataSource choose *Create InfoPackage* and in the presented dialog enter the name **FIL 0 - Delta Initialization GR##**. Choose  icon..
- b) On the *Data Selection* tab enter the following value:

Field	Value
<i>CALYEAR (Calendar Year)</i>	2001

- c) On the *Processing* tab you will see that the option **Only PSA** is not available for change. In BI, InfoPackages only extract data to the PSA table.
- d) On the *Update* tab set the Update Mode **Initialize Delta Process – Initialization with Data Transfer**.
Then choose the *Continue (Enter)*  icon.
- e) Ordinarily, the InfoPackage would not be scheduled on the *Schedule* tab, but the InfoPackage would be included in a Process Chain. In order to save time, in this exercise, we will not utilize a Process Chain, so on the *Schedule* tab, make sure the option **Start Data Load Immediately** is selected and then click the button *Start*.
- f) Within InfoPackage Maintenance (Scheduler), choose the *Monitor*  icon.
- g) On the tab page, choose *Detail* and the pushbutton *Refresh Details*  until the load process is finished. All lights should be green.

Continued on next page

Task 13:

Create a Data Transfer Process to load PSA data into your DataStore Object. Check the uploaded data in the DataStore Object.

1. To load the data from the PSA table of DataSource **BW350_FF_NEWGR##** into the associated DataStore Object **B350FF##**, you have to create a Data Transfer Process. On the *Extraction* tab ensure that the *Extraction Mode* is set to Delta. Execute the Data Transfer Process.
 - a) Expand the tree under your DataSource **BW350_FF_ADDGR##** until you locate the DataStore Object **B350FF##**.
 - b) In the context menu for the DataStore Object **B350FF##**, choose *Create Data Transfer Process*. Enter the following information in the *Creation of Data Transfer Process* screen.

Field	Value
<i>Object Type</i>	DataSource
<i>DataSource</i>	BW350_FF_NEWGR##
<i>Source System</i>	I_EXTERN

Choose the *Continue (Enter)*  icon.

- c) In the Data Transfer Process, go to the *Extraction* tab and choose the *Delta* option from the dropdown box if it is not already displayed. Choose the *Activate*  icon to activate your Data Transfer Process.
 **Note:** You do not need to initialize the delta process explicitly for the delta transfer.
- d) In the Data Transfer Process, go to the *Execute* tab and choose the *Execute*  icon.
 **Note:** If you do not include the Data Transfer Process in a Process Chain, you can execute it on the *Extraction* tab.

Continued on next page

2. Check the status of the extraction and the contents of the DataStore Object in the *DTP Monitor* respectively within *DataStore Object Administration*.



Note: The uploaded data will not be activated automatically in the DataStore Object **B350FF##**.

- a) From the Data Transfer Process **BW350_FF_NEWGR## / I_EXTERN -> B350FF##**, choose the *Monitor* icon to check the status of the extraction in the *DTP Monitor*. If necessary, choose the button *Refresh Details* until the upload is finished. All lights should be green. Go back to the Data Warehousing Workbench using the *Back* button.
- b) From the context menu of your DataStore Object **B350FF##** choose *Manage* and go to the *Contents* tab and check the contents of the *Activation Queue (New Data)*, *Active Data and Change Log* tables in succession. The Activation Queue should have three records and the Active Data and Change Log tables should be empty.
- c) Go to the *Requests* tab and choose the *Activate* icon.
- d) Select the RequestID by clicking the empty square to the left of the RequestID and choose the *Start* icon.
- e) Go to the *Contentstab* and check the contents of the *Activation Queue (New Data)*, *Active Data and Change Log* tables in succession again. The Activation Queue should be empty and the Active Data and Change Log tables should each have six records.

Task 14:

Create an InfoPackage for use in loading delta data to the PSA table of the DataSource.

1. . Create a second InfoPackage for your DataSource **BW350_FF_NEWGR##** in order to request the delta by loading the second CSV-File **T_BW350_DELTA_LOAD** with delta data into the PSA table of your DataSource. Use **FIL0 - Delta GR##** as InfoPackage description.

Continued on next page

Set the *Update Mode* on the *Update* tab to **Delta Update**. On *Schedule* tab start the data load immediately. Check your data load in the *Data Load Monitor*.

- a) Locate your DataSource **BW350_FF_NEWGR##**. From the context menu of your DataSource choose *Create InfoPackage* and in the presented dialog enter the name **FIL0 - Delta GR##**. Choose *Save*  icon.
- b) On the *Extraction* tab, enter the following value for *Name of File* and accept the defaults:

<i>Name of the File</i>	<Individual Path> T_BW350_DELTA_LOAD.CSV
-------------------------	---

- c) On the *Processing* tab you will see that the option **Only PSA** is not available for change. In BI, InfoPackages only extract data to the PSA table
- d) On the *Update* tab set the Update Mode **Delta Update**.
- e) Ordinarily, the InfoPackage would not be scheduled on the *Schedule* tab, but the InfoPackage would be included in a Process Chain. In order to save time, in this exercise, we will not utilize a Process Chain, so on the *Schedule* tab, make sure the option **Start Data Load Immediately** is selected and then click the button *Start*.
- f) Within InfoPackage Maintenance (Scheduler), choose the *Monitor*  icon.
- g) On the tab page, choose *Detail* and the pushbutton *Refresh Details*  until the load process is finished. All lights should be green.

Task 15:

Load the delta from the PSA table into your DataStore Object **B350FF##** by executing the existing Data Transfer Process again. Check the uploaded data in the DataStore Object

1. Execute the Data Transfer Process **BW350_FF_NEWGR## / I_EXTERN -> B350FF##** again.
 - a) Expand the tree under your DataSource **BW350_FF_NEWGR##** until you locate the DataStore Object **B350FF##**.
 - b) Find the Data Transfer Process **BW350_FF_NEWGR## / I_EXTERN -> B350FF##** and go to the Execute tab and choose the *Execute*  icon.

Continued on next page

2. Check the status of the extraction and the contents of the DataStore Object in the *DTP Monitor* respectively within *DataStore Object Administration*.



Note: The uploaded data will not be activated automatically in the DataStore Object **B350FF##**.

- a) From the Data Transfer Process **BW350_FF_NEWGR## / I_EXTERN -> B350FF##**, choose the *Monitor* icon to check the status of the extraction in the *DTP Monitor*.
If necessary, choose the button *Refresh Details* until the upload is finished. All lights should be green.
- b) Go back to the Data Warehousing Workbench using the *Back* button.
- c) From the context menu of your DataStore Object **B350FF##** choose *Manage* and go to the *Contents* tab and check the contents of the *Activation Queue (New Data)*, *Active Data and Change Log* tables in succession. The Activation Queue should contain the three delta records (Quantity = 5.000) The Active Data and Change Log tables should be unchanged (Quantity = 10.000).
- d) Go to the *Requests* tab and choose the *Activate* button.
- e) Select the RequestID by clicking the empty square to the left of the Request ID and choose the *Start* icon.
- f) Go to the *Contents* tab and check the contents of the *Activation Queue (New Data)*, *Active Data and Change Log* tables in succession again. The Activation Queue should be empty The Active Data should contain three records (Quantity = 5.000). The Change Log should contain 9 records (3x New Image, 3x Before Image, 3x After Image).



Lesson Summary

You should now be able to:

- Explain the basic prerequisites of extraction using the file interface.
- Create your own file system
- Use enhanced functions for preparing, testing and loading files using the file adapter
- Decide, based on your knowledge of the source data, which process should be used during data loading
- Use a DataStore object to run delta load processes for the file interface



Unit Summary

You should now be able to:

- Explain the basic prerequisites of extraction using the file interface.
- Create your own file system
- Use enhanced functions for preparing, testing and loading files using the file adapter
- Decide, based on your knowledge of the source data, which process should be used during data loading
- Use a DataStore object to run delta load processes for the file interface

Unit 6

Data Acquisition with DB Connect

Unit Overview

In this unit, we will first describe the reasons for performing data transfer with DB connect. This unit then introduces the DB Connect architecture and the prerequisites for DB Connect. You will see how DB Connect source systems and DataSources are defined.



Unit Objectives

After completing this unit, you will be able to:

- Describe the concept behind DB Connect and its architecture
- Name the database management systems that support DB Connect technology
- Set up for using DB Connect
- Create a database source system
- Define a DB Connect DataSource

Unit Contents

Lesson: Data Acquisition with DB Connect	348
Exercise 11: Data Transfer With DB Connect	363

Lesson: Data Acquisition with DB Connect

Lesson Overview

This lesson explains the concept behind DB Connect and its architecture. It also presents the database management systems that support data transfer with DB Connect. Furthermore, this lesson describes the procedure for creating a database system as a source system and defines DB Connect DataSources.



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the concept behind DB Connect and its architecture
- Name the database management systems that support DB Connect technology
- Set up for using DB Connect
- Create a database source system
- Define a DB Connect DataSource

Business Example

Your company extracts data from external systems using flat files. The data volume from the various external systems has increased continually in the recent past, making management of the jobs for flat file extraction extremely difficult. You can optimize this process by using DB Connect to extract data directly from an external system.

DB Connect

The following graphic shows how the DB Connect technology provides another alternative for extracting data from an external system.

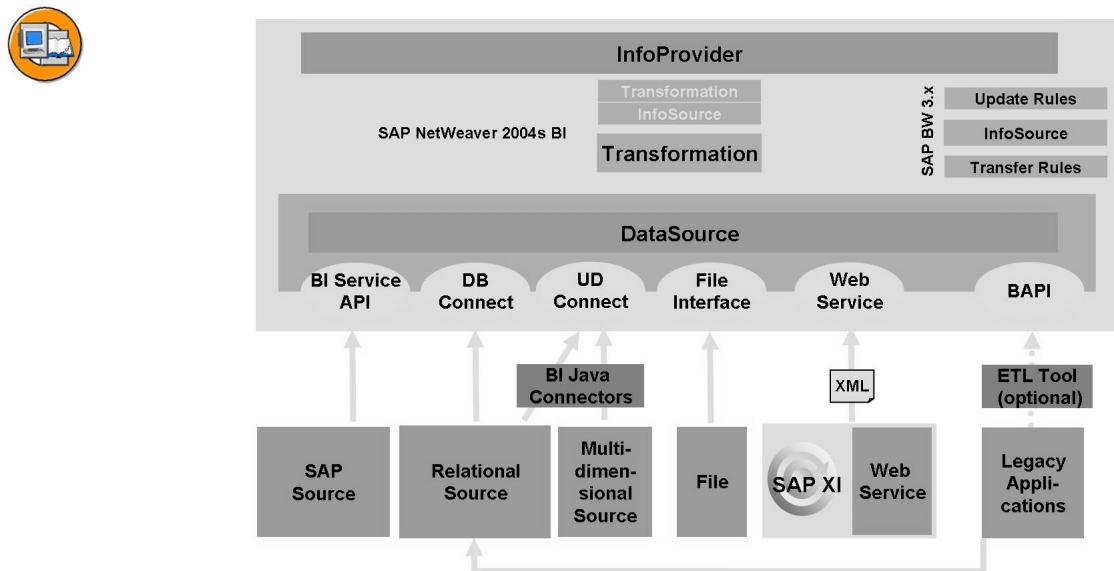


Figure 135: Data Staging Architecture

With DB Connect, SAP provides another option for extracting data from external systems. DB Connect enables you to extract data from tables/views in database management systems (DBMS) which, in addition to having the standard connection, are also connected to the BI system, and transfer it to the BI system. You can use tables and views from the SAP-supported DBMS for data transfer. You can then use a DataSource to make the data available to BI and transfer it to BI InfoProviders using the usual data acquisition process.

To load the data from a SAP-supporting DBMS into BI using DB Connect, you have to:

- Connect a database to the BI system as a source system, thus providing direct access to the external, relational DBMS (RDBM).
- Make the metadata for the table/view of the external RDBMS known to the BI using the definition of a DataSource.

The following graphic shows an example of the application of DB Connect

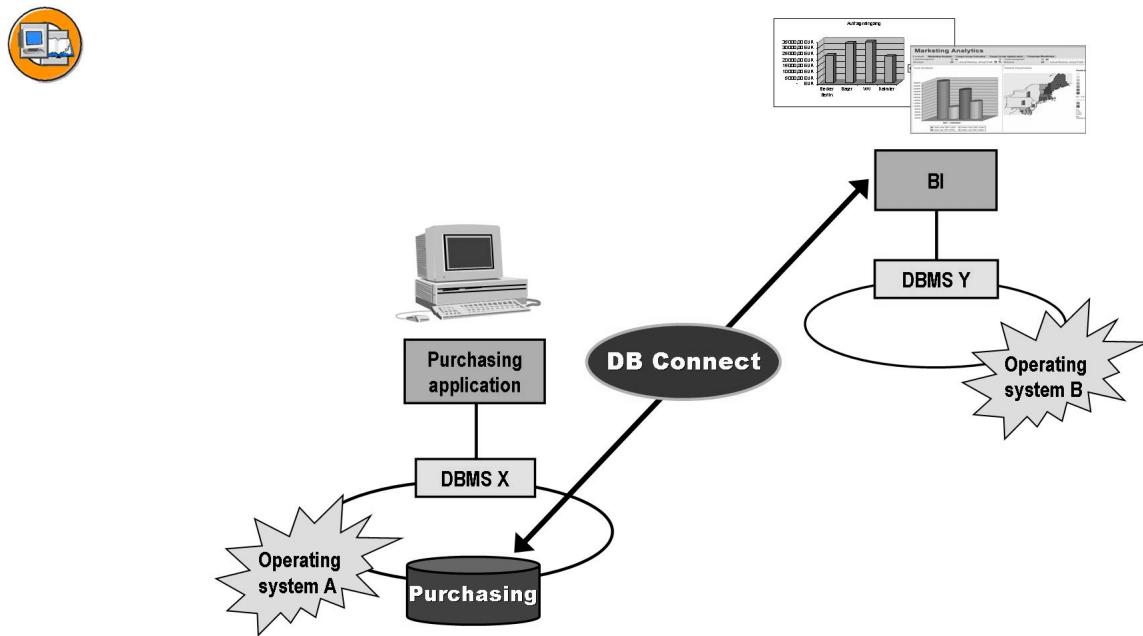


Figure 136: DB Connect - Example

The purchasing application runs on a system that is based on a DBMS X. To analyze the data from the purchasing application, the data has to be loaded into a BI system based on a DBMS Y. DBMS Y can be If the DBMS Y is the same as DBMS X, you do not need to install the database-specific client and the database-specific DBSL. The functionality of the DB Connect enables you to connect the DBMS for the purchasing application and then extract data from the DB tables or views and transfer it into BI.

DB Connect architecture

When starting a BI application server, the SAP kernel opens a connection to the database on which the BI system is running. This connection is designated as the **default** or **standard connection** because all open SQL and native SQL statements automatically refer to this connection. If you want to set up another connection to the DBMS - of the same type or a different type - in addition to the default connection, you can use DB Connect functionality, namely the SAP technology component **Multi-Connect**.

The following graphic presents the architecture of DB Connect, which will be described in more detail in the following.

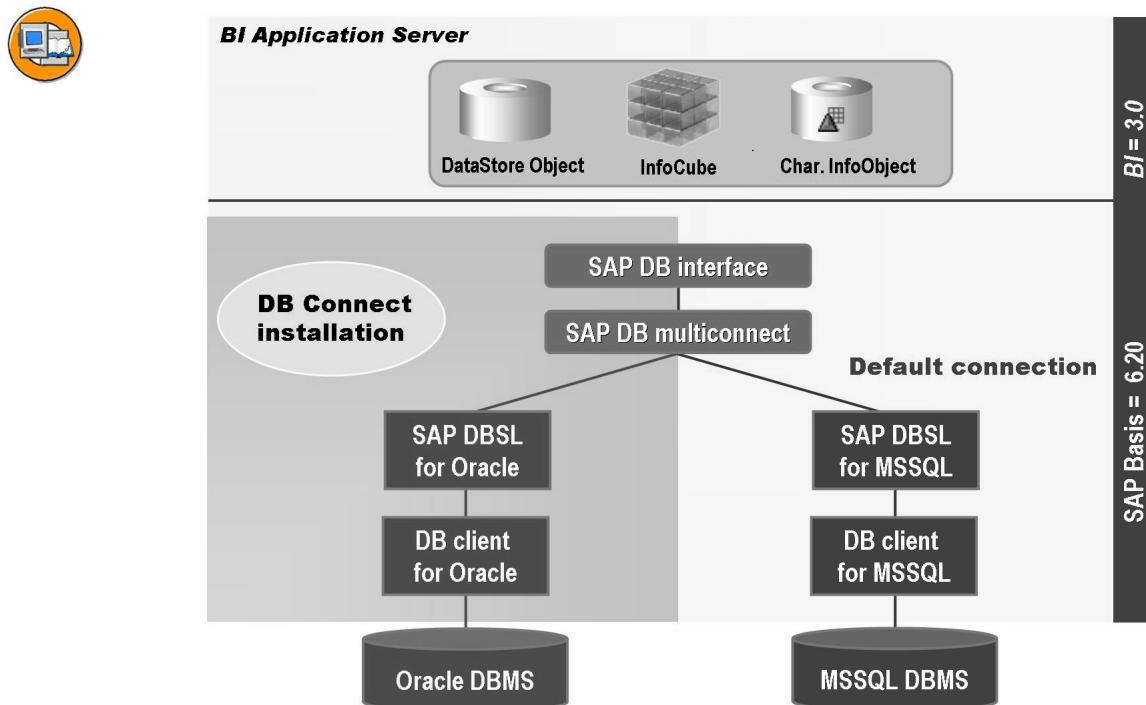


Figure 137: DB Connect architecture

Database Management Systems that Support DB Connect

In principle, only *BI* application servers for which SAP has released a DBSL for the *BI* DBMS and the source DBMS can be supported on the operating system versions for DB Connect.



- **Max DB [previously SAP DB]** (SAP Note: 520647)
- **Informix** (SAP Note: 520496)
- **Microsoft SQL Server** (SAP Note: 512739)
- **Oracle** (SAP Note: 518241)
- **IBM DB2/390** (SAP Note: 523552)
- **IBM DB2/400 I Series** (SAP Note: 523381)
- **IBM DB2 UDB** (SAP Note: 523622)

Figure 138: DB Connect Supported DBMS

- **Max DB [previously SAP DB]**

Requirements for the source DBMS:

- Max DB [previously SAP DB] (\geq 7.2.5 Build 3)

Requirements for the BI DBMS:

- BI (\geq 3.0B)
- SAP WAS (\geq 6.20)
- Max DB [previously SAP DB] Client (\geq SAP DB 7.3.1)

- **Informix**

Requirements for the source DBMS:

- All IDS versions of the 7 and 9 series that are part of IBM/Informix maintenance

Requirements for the BI DBMS:

- BI (\geq 3.0B)
- SAP WAS 6.20, SP 2
- SAP WAS 6.20, SP 2 – Informix DB Client (\geq I-Connect 2.70)

- **Microsoft SQL Server**

Requirements for the source DBMS:

- Windows NT – Application server
- Microsoft SQL Server (7.0 or later) and Microsoft SQL Server (2000 or later)

Requirements for the BI DBMS:

- BI (3.0B or later)
- SAP WAS 6.20, SP 2
- Application server: Windows NT
- DB Client (Microsoft SQL Server 7.0 or later)

- **Oracle**

Requirements for the source DBMS:

- Oracle (8.1.7.3 or later)

Requirements for the BI DBMS:

- BI (\geq 3.0B)
- SAP WAS 6.20, SP 2
- DB Client (Oracle \geq 8.1.7.3, delivered with SAP WAS 6.20)

- **IBM DB2/390**

Requirements for the source DBMS:

- DB2/390 (\geq V6)

Requirements for the BI DBMS:

- BI (\geq 3.0B)
- SAP WAS 6.20, SP 4
- DB Client (ICLI): (\geq 6.20)
- **IBM DB2/400**
 - Requirements for the source DBMS:
 - DB2/400(\geq V4R5)
 - Both EBCCDIC and ASCII data can be read
 - Requirements for the BI DBMS:
 - BI (\geq 3.0B)
 - SAP WAS 6.20, SP 3
 - Application server: Windows NT
 - DB Client: As of IBM Client Access Express and XDA (\geq V5R1) and \geq Release of the source DB
- **IBM DB2 UDB**
 - Requirements for the source DBMS:
 - DB2 for Unix and Windows V7.1 and higher
 - Only use FixPaks permitted by SAP (SAP Note: 200225)
 - Requirements for the BI DBMS:
 - BI (\geq 3.0B)
 - SAP WAS 6.20, SP 3
 - DB Client: DB2 UDB Run-time Client for Unix and Windows V7.2 and higher



Hint: The information listed above could be subject to change. It is therefore important that you consult the SAP Notes listed above.

Setup for DB Connect – Data and Metadata Staging

Before you can use the DB Connect functions, you must prepare as follows::

- Install the DB client
- Install an SAP-specific database shared library

First you need to install a **DB client**, that is, another database-specific client for the respective source DBMS on the BI application server, as long as this DBMS is different from the DBMS used by BI. A license for this DB client can usually be obtained from the manufacturer of the database. You can get more information about the DB client from the respective database manufacturer.

Furthermore, an SAP-specific DBSL (Database Shared Library) must be installed on the BI application server for the database to be connected. You can obtain this DBSL from SAP. The data-dependent part of the DB interface is located in its own library that is dynamically connected to the SAP kernel. In addition to the DBSL mentioned, the library also contains further libraries from the respective database manufacturer that can either be connected to the DB library statically or dynamically. For more information, see “Installation of the Database Shared Library (DBSL)” in the general documentation and in SAP Note 400818.

Before the correct data from the applications from external RDBMS can be loaded into BI, the metadata, which is the table or view field information in BI has to exist in the form of a DataSource. With DB Connect you can use the connection to the external system supported by SAP as a source system connection for the BI system. You can use DB tables and views from the RDBMS database catalog to generate DataSources. DB Connect also includes the mapping from database data types on ABAP data types. These DataSources have the task of making the metadata known to BI. Then the usual data acquisition process of BI comes into play.

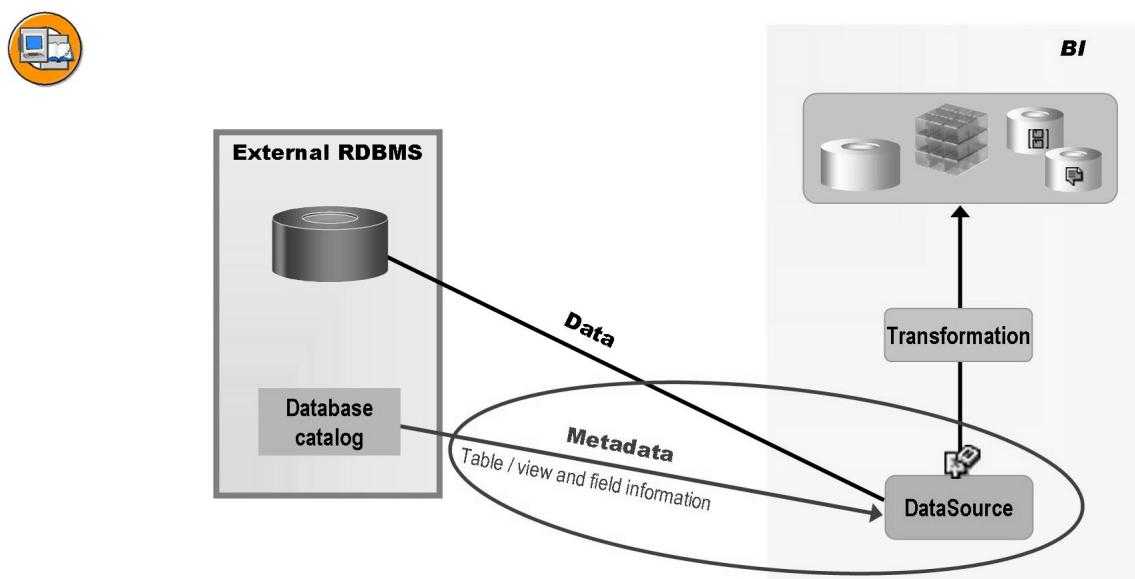


Figure 139: DB Connect Data and Metadata Acquisition

DB Connect definition of database source systems

Before a database connection can be opened, all connection data that is required for identification of the source database and for authentication against the database must be made known to the runtime environment. You must specify the connection data for every database connection that is also to be built in addition to the SAP default connection.

In the first step, you have to create the database source system in the Data Warehousing Workbench:

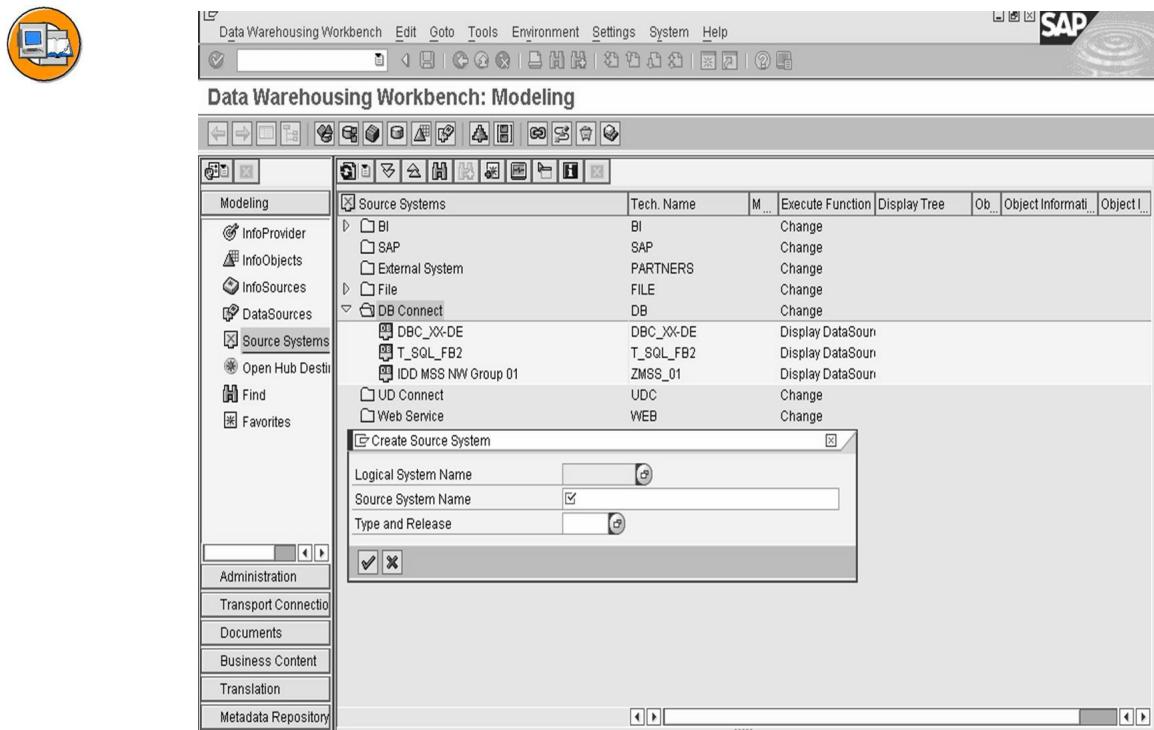


Figure 140: Define DB Connect DataSource (1)

The connection data (see the graphic below) that is used to identify the source database and authenticate this database is determined here.

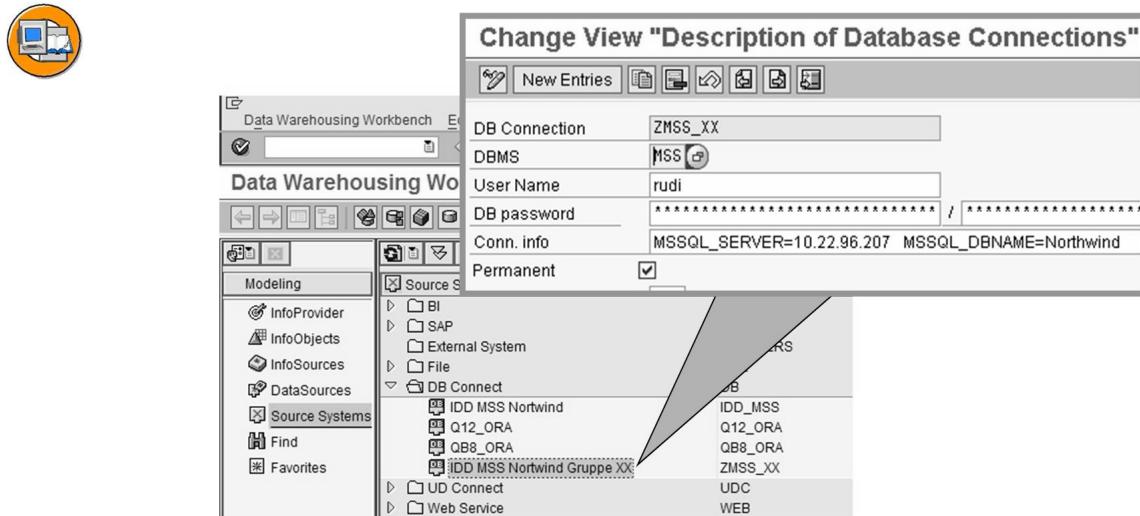


Figure 141: DB Connect Connection Parameters

The example in the graphic above illustrates the information format for a Microsoft SQL database. The fields in the screen shown above are explained in the following text:

DB Connection

Enter the logical system name here.

DBMS

Use the selection list to choose the database management system (DBMS). In this way, you determine which type of database platform you are dealing with for this connection.

Possible database platforms:

- ADA: SAP DB (ADABAS/D)
- DB2: DB2 UDB for OS/390
- DB2: DB2 UDB for AS/400
- DB6: DB2 UDB for Unix and Windows
- INF: Informix
- MSS: Microsoft SQL Server
- ORA: Oracle

User name

Here you enter the database user whose name is to be used to open the connection.

Password

Here you enter the DB password of the user twice for authentication against the database when establishing the connection.

Conn. info

Here you enter the technical information on opening a database connection. This information, which is needed to build a connection in *NATIVE SQL*, depends on the database platform and generally includes the database name and the host on which the database runs. The string tells the respective client library to which database the connection should be established.

Permanent

The *Permanent* indicator is needed when an open database connection is interrupted. This can occur when the database itself or the network connection breaks down. The SAP workflow tries to re-establish the interrupted connection irrespective of this indicator. If the attempt fails, the system responds as follows:

1. “DB reconnection type is not *permanent* (DBCON_RECO = BLANK)”
The error is ignored and the transaction that was called is started. If this transaction accesses a connection that does not exist, the transaction is terminated.
2. “DB reconnection type is not *permanent* (DBCON_RECO = BLANK)”
After the first connection error, the system checks whether a new connection is possible before each transaction. If not, the transaction is not started. This is valid irrespective of whether the current transaction were to access this special connection or not. The SAP system only functions completely when all permanent DB connections are restored.
The standard connection established at the start has the implicit DB



Hint: The indicator should be set if the opened DB connection is indispensable or is accessed quite often.

Connection limit

Set the maximum number of simultaneous data connections that can be opened for the logical connection name in a running SAP transaction. Any further connection requests (CONNECT) will be declined. The initial value 0 has the special meaning that the maximum number of connections will be allowed, up to 255. A DB connection can not be used by multiple SAP work processes simultaneously; each work process needs its own (private) DB connections. With many DB connections and many work processes, this can lead to resource problems on the DB server. The parameter is also suited for this example:

- DBCON_MAX_CONNECTIONS = 0
maximum of 255 DB connections
- DBCON_MAX_CONNECTIONS = 1
maximum of 1 DB connection
- DBCON_MAX_CONNECTIONS = 5
maximum of 5 DB connections

Connection optimum

Sets the number of optimal DB connections. If this number is exceeded in an SAP transaction, the work process makes sure that the connections beyond that number are closed after the transaction has ended. This is an automatic DISCONNECT. The initial value 0 has special significance: If DBCON_MAX_CONNECTIONS is also 0, the optimum is extended to the maximum value for the value range of the variables (255). A DB connection can not be used by multiple SAP work processes simultaneously; each work process needs its own (private) DB connections. With many DB connections and many work processes, this can lead to resource problems on the database server. The parameter is able to set a soft limit for the consumption of resources during productive operation; this limit can be exceeded occasionally. The following is an example:

- DBCON_MAX_CONNECTIONS = 0, DBCON_OPT_CONNECTIONS = 0
 - maximum/optimal 255 DB connections
- DBCON_MAX_CONNECTIONS = 1, DBCON_OPT_CONNECTIONS = 0
 - maximum 1 DB connection, optimally no DB connection
- DBCON_MAX_CONNECTIONS = 5, DBCON_OPT_CONNECTIONS = 2
 - maximum 5 DB connections, optimally 2 DB connections

With the connection data listed above, you have established the connection to a database source system. You will find the entry for this in the source system of the Data Warehousing Workbench. Now you can define DataSources for the tables or views on the database.

Definition of DB Connect DataSource

In order to be able to transfer data from a database source system, you have to make metadata - that is, the table/view and field information - known to BI using DataSources.

Definition of a DB Connect (DBC) DataSource:

In the first step, in the context menu for the DB Connect source system created previously, choose the function *Create DataSource*.

On the *Extraction* tab, you select the Table Adapter as shown below.

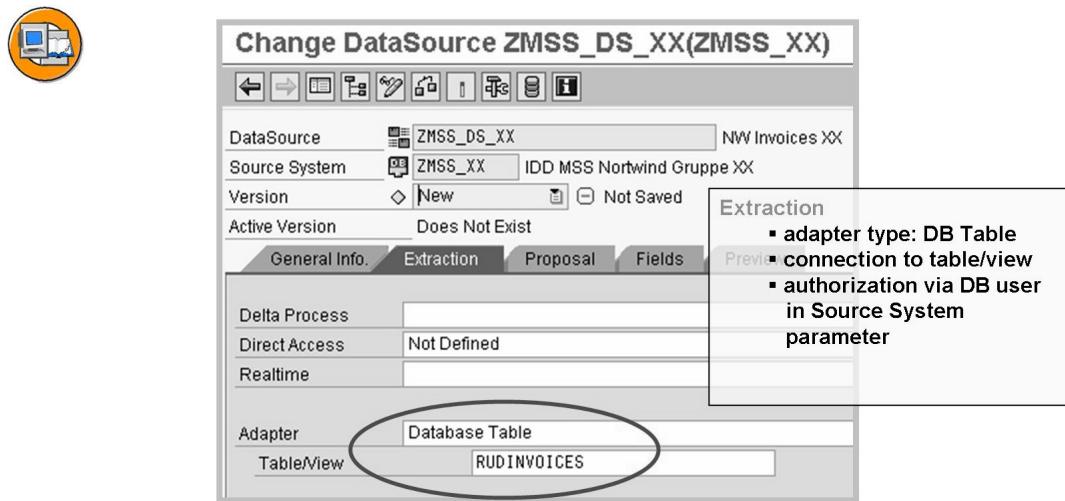


Figure 142: DB Connect DataSource Table Adapter

From the database source system, select a table or view for data extraction by choosing *Table/View*. On the next screen, select whether you want tables and/or views to be available for selection, enter the required data under *Table/View* and choose *Execute*. The database connection is established and the database tables are read.

The *Choose DB Object Names* screen appears. On this screen the system displays, in accordance with your selections, the tables and views that are stored in the database schema of the database user for which the connection has been established. The technical name, type and database schema for a table or view are displayed.

→ **Note:** Make sure that you use only tables and views in the extraction whose technical names consist solely of upper case letters, numbers, and underscores (_). Using other characters can lead to problems.

Go to the *Proposal* tab page. The fields of the table or view are displayed here. The overview of the database fields tells you which fields are key fields, the length of the field in the database compared with the length of the field in the ABAP data dictionary, as well as the field type in the database and the field type in the ABAP dictionary as shown below. It also gives you additional information to help you check the consistency of your data.



Change DataSource URTEST(IDD_MSS)

DataSource: URTEST | DB Con Example
Source System: IDD_MSS | IDD MSS Nortwind
Version: New | Not Saved
Active Version: Does Not Exist

General Info. Extraction Proposal Fields Preview

Extractor	Database Table												
Co	P.	Database Field	Datasource Field	Type in Dat	Length	Decimals	Pri	Sec	Type in Dat	Length	Precision	Scale	Zero V.
✓	1	CustomerID	CUSTOMERID	CHAR	10				nchar	10	5	0	
✓	2	OrderID	ORDERID	INT4	4				int	4	10	0	X
✓	3	OrderDateV	ORDERDATEV	CHAR	8				varchar	8	8	0	
✓	4	ProductID	PRODUCTID	INT4	4				int	4	10	0	X
✓	5	UnitPriceV	UNITPRICEV	DEC	17	2			decimal	9	17	2	
✓	6	QuantityV	QUANTITYV	DEC	17	3			decimal	9	17	3	

Proposals

- only visible during design time
- meta data information of connected table/view as proposal
- automatic data type conversion
- can be copied into the field list tab strip

**Figure 143: DB Connect DataSource Proposal of Fields**

On the Field List tab, you edit the fields that you transferred to the field list of the DataSource from the Proposal tab page as shown below.

Field list

Defines the DataSource structure
Can be enhanced with Template InfoObjects

DataSource: DBCMSS_INV02 | NW Invoices
Source System: IDD_MSS | IDD MSS Nortwind
Version: A | Edited Version

General Info. Extraction Proposal Fields Preview

Pos.	Field	Descript.	D.	T	Template I...	Data type	Lngth	Deci...	extern...	L	K	Con.
1	CUSTOMERID	CustomerID	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		CHAR	10	0	10	<input type="checkbox"/>	<input type="checkbox"/>	
2	ORDERID	OrderID	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		INT4	4	0	11	<input type="checkbox"/>	<input type="checkbox"/>	
3	ORDERDATEV	OrderDateV	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	0CALDAY	CHAR	8	0	8	<input type="checkbox"/>	<input type="checkbox"/>	
4	PRODUCTID	ProductID	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		INT4	4	0	11	<input type="checkbox"/>	<input type="checkbox"/>	
5	UNITPRICEV	UnitPriceV	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		DEC	17	2	19	<input type="checkbox"/>	<input type="checkbox"/>	
6	QUANTITYV	QuantityV	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		DEC	17	3	19	<input type="checkbox"/>	<input type="checkbox"/>	

Figure 144: DB Connect Field List

- Under Transfer, specify the decision-relevant DataSource fields that should be available for extraction and transferred to BI.
 **Note:** Note that the technical field names of tables and views can be no longer than 16 characters and must only contain upper case letters, numbers, and underscores (_). Using other characters can lead to problems. You cannot use fields with reserved field names, such as COUNT. Fields that do not comply with these restrictions are not available for extraction.
- If required, change the values for the key fields of the source. These fields are generated as a secondary index in the PSA. This is important in ensuring good performance for data transfer process selections, in particular with semantic grouping.
- Specify whether the source provides the data in the internal or external format.
- If you choose an External Format, ensure that the output length of the field (external length) is correct. Change the entries, as required.
- If required, specify a conversion routine that converts data from an external format into an internal format.
- Select the fields for which you want to be able to set selection criteria when you schedule a data request using an InfoPackage. Data for this type of field is transferred in accordance with the selection criteria specified in the InfoPackage.
- Choose the selection options (such as EQ, BT) that you want to be available for selection in the InfoPackage.
- Under Field Type, specify whether the data to be selected is language-dependent or time-dependent as required. If you did not transfer the field list from a proposal, you can define the fields of the DataSource directly. Choose Insert Row and enter a field name. You can specify InfoObjects in order to define the DataSource fields. Under Template InfoObject, specify InfoObjects for the fields of the DataSource. This allows you to transfer the technical properties of the InfoObjects into the DataSource field. Entering InfoObjects here does not equate to assigning them to DataSource fields. Assignments are 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.

Exercise 11: Data Transfer With DB Connect

Exercise Objectives

After completing this exercise, you will be able to:

- Connect a SAP BW system to other database source systems
- Use previously defined DB tables/views to define DB Connect DataSources

Business Example

In this scenario you will stage data relating to your employees from the Northwind database to SAP BW using a DB Connect DataSource. The data is already provided by predefined views in the Northwind database.

Task:

1. Define a DB Connect DataSource **GR##DBDS** for master data attributes in your Application Component **ZT_BW350_GR##**, to provide all the fields of the data base view **EMPLOYEE##** for the data staging process in BI.

To do this, use the database source system **T_SQL_F2**.

Use the following descriptions for the DataSource:

<i>Short Description</i>	DB DatsSource ##
<i>Medium Description</i>	DB DataSource Group ##
<i>Long Description</i>	DB DataSource Group ##

Solution 11: Data Transfer With DB Connect

Task:

1. Define a DB Connect DataSource **GR##DBDS** for master data attributes in your Application Component **ZT_BW350_GR##**, to provide all the fields of the data base view **EMPLOYEE_##** for the data staging process in BI.

To do this, use the database source system **T_SQL_F2B**.

Use the following descriptions for the DataSource:

<i>Short Description</i>	DB Datasource ##
<i>Medium Description</i>	DB DataSource Group ##
<i>Long Description</i>	DB DataSource Group ##

- a) In the **Data Warehousing Workbench**: use the menu, *Modeling* → *Source Systems*

From the *Context Menu* of **T_SQL_F2B**, choose *Display DataSource Tree*

- b) In the DataSource tree, find the Application Component **ZT_BW350_GR##**, and from the context menu choose the option *Create DataSource* and enter the following values in the dialog *Create DataSource*:

<i>DataSource</i>	GR##DBDS
<i>Source System</i>	T_SQL_F2B
<i>DataSource Type</i>	Master Data Attributes

Choose the *Continue (Enter)* icon to continue.

- c) Enter descriptions on the *General* tab as follows:

<i>Short description</i>	DB Datasource ##
<i>Medium description</i>	DB DataSource Group ##
<i>Long description</i>	DB DataSource Group ##

- d) On the *Extraction* tab use the selection list function for the field *Table/View* to choose a database table or view.

Continued on next page

In the following dialog box, select the checkbox **Select Views** and choose the *Execute*  icon to continue and you will be presented with a list of database views from the source system.

Select the database view **EMPLOYEE_##** and choose the *Continue (Enter)*  icon.

- e) On the *Proposal* tab, you will find all the fields of your view are chosen by default along with a proposed type conversion.
Keep all the fields marked and proceed to the field list of the *Fieldtab*.
- f) In the *Fieldtab* you can bring in the latest changes or adjustment of the DataSource structure. Template InfoObjects can be referenced as well. For this example there are no additional changes necessary.
Click the *Save*  icon and then the *Activate*  icon.
- g) In the *Preview* tab, you can look at the data from the database view by choosing the *Read Preview Data* button. Now all the records should be visible.



Lesson Summary

You should now be able to:

- Describe the concept behind DB Connect and its architecture
- Name the database management systems that support DB Connect technology
- Set up for using DB Connect
- Create a database source system
- Define a DB Connect DataSource



Unit Summary

You should now be able to:

- Describe the concept behind DB Connect and its architecture
- Name the database management systems that support DB Connect technology
- Set up for using DB Connect
- Create a database source system
- Define a DB Connect DataSource

Unit 7

Data Acquisition with Universal Data Integration

Unit Overview

In this unit, first the concept and architecture of universal data integration (UDI) are described. The UDI components are introduced, with a focus on the key *UD Connect* components, which enable both persistent and transient data staging in BI. Furthermore, the UD Connect architecture is described along with the required connection configurations that allow you to use UD Connect. The individual steps for defining a UD Connect DataSource are also described.



Unit Objectives

After completing this unit, you will be able to:

- Describe the concept and architecture of UDI
- Name the key components of UDI
- Get an overview of the functions of the BI Java Connectors
- Outline the UD Connect architecture
- Explain the significance of the BI Java SDK

Unit Contents

Lesson: Universal Data Integration.....	370
Procedure: Creating DataSources for UD Connect.....	386

Lesson: Universal Data Integration

Lesson Overview

This lesson describes the concept and architecture of universal data integration. It provides an initial overview of the UDI components, focusing on the key UDI component **UD Connect**, which enables both persistent and transient data staging in BI.

In addition, this lesson describes the UD Connect architecture and the connection configurations you require in order to use UD Connect. Then the individual steps for defining a UD Connect DataSource are described.



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the concept and architecture of UDI
- Name the key components of UDI
- Get an overview of the functions of the BI Java Connectors
- Outline the UD Connect architecture
- Explain the significance of the BI Java SDK

Business Example

Your company would like to establish a direct open connection to your Microsoft Access database. In considering DB Connect technology, you have decided that this technology is not a suitable option for your particular database. You need a data staging option that can support a wider range of source systems. The options provided by the SAP BI connection are considerably enhanced by universal data integration so that almost all data sources can be directly connected to SAP BI.

Universal Data Integration: Concept and Architecture

The new development of UDI technology for BI was mainly based on market demand for improved integration options between BI and existing, heterogeneous system landscapes. There was a demand for scenarios in which the data staging can be expanded to a broader range of data sources and the restrictions on the DB Connect overcome without making the use of third-party ETL tools a requirement. Access to both multidimensional and relational DBMs systems must also be possible. Transient scenarios (direct access using VirtualProviders) should also be possible, such as conventional persistent staging of data in BI.

The following graphic provides an overview of the UDI architecture.

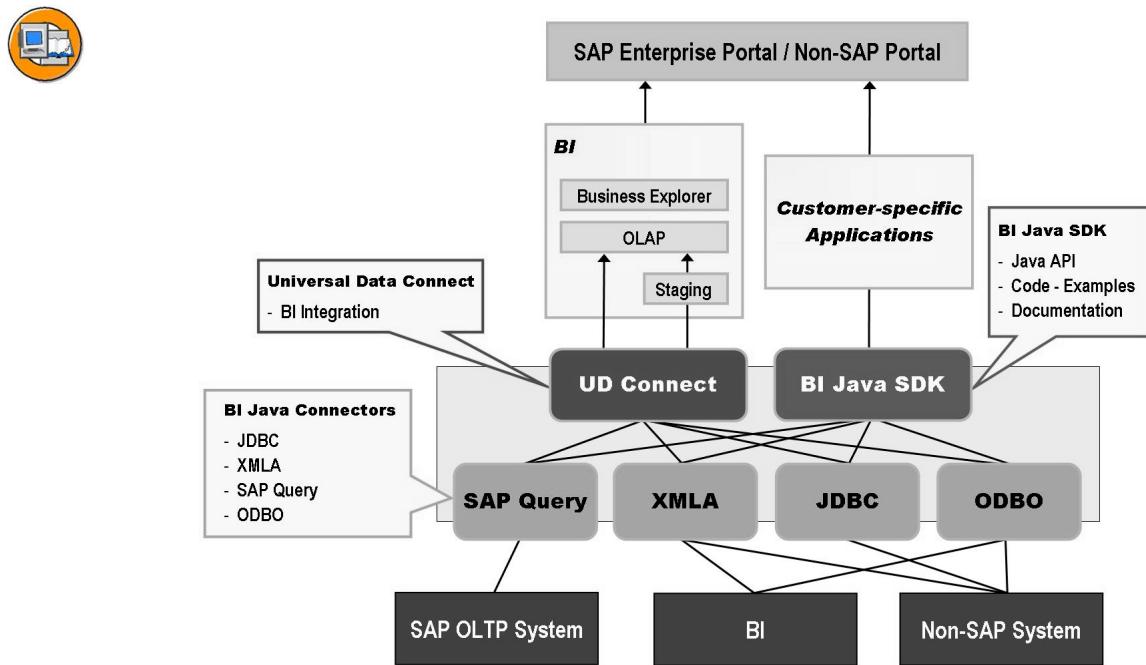


Figure 145: UDI Architecture

BI Java Connectors

The **BI Java Connectors** form the basis on the J2EE server of the SAP Web Application Server (SAP WAS). The following connector types are oriented to the usual industry standards for interoperability:

- BI Java Database Connectivity (JDBC) Connector
- BI OLE DB for OLAP (ODBO) Connector
- BI XML for Analysis (XMLA) Connector
- BI SAP Query Connector

Based on these connectors, the **Universal Data (UD) Connect component** includes the Java-based implementation of BI. Interestingly enough, both persistent and transient data staging is supported here. (This is indicated in the graphic with the two arrows “UD Connect to BI Staging” and “UD Connect to BI OLAP.”)

The **BI Java Software Development Kit (SDK) component** enables the development of customer-specific applications, in certain circumstances with the aim of publishing acquired data in the SAP Enterprise Portal.

The following will explain the three main components of UDI architecture – BI Java Connectors, UD Connect, and BI Java SDK – in more detail.

BI Java Connectors

As the graphic above illustrates, BI Java Connectors are available that can be used in connection with BI Java SDK or UD Connect to extract data from SAP OLTP systems, BI systems and external systems. The prerequisite is that you have installed SAP WAS J2EE Engine with the BI Java components. For more information on this subject, go to the SAP Service Marketplace at <http://www.service.sap.com/instguide>.

The available BI Java Connectors are listed in the following graphic.



- **BI JDBC Connector**
→ For connecting to relational JDBC data sources
- **BI ODBO Connector**
→ For connecting to OLE DB for OLAP-capable data sources
- **BI SAP Query Connector**
→ For connecting to data from operational SAP applications
- **BI XMLA Connector**
→ For connecting to XMLA-capable data providers, such as SAP BW

Figure 146: Available BI Java Connectors

BI JDBC Connector

Sun Microsystem's JDBC (Java Database Connectivity) is the standard Java API for Relational Database Management Systems (RDBMS). With the JDBC connector you can connect applications that were created with Java SDK with more than 200 JDBC drivers. The following data sources are supported: Teradata, Oracle, Microsoft SQL Server, Microsoft Access, DB2, Microsoft Excel and text files (such as CSV files). You can also use the BI JDBC Connector to stage data from these data sources in BI using UD Connect (see the “UD Connect” section). For more information on JDBC from Sun Microsystems, see <http://java.sun.com/products/jdbc>.

The BI JDBC Connector enables the following functions for the JDBC drivers:

- Universal connection management that is integrated into the user management in the SAP Enterprise Portal.
- A query model that is independent of the SQL dialect of the underlying data source.
- A unified metadata service via implementation of JMI functions that are based on CWM.



Hint: The Common Warehouse Model (CWM) is a standard OMG that provides a framework for representing metadata in data warehousing, business intelligence, knowledge management, and portal technologies.

The Object Management Group (OMG) Java Metadata Interface (JMI) Specification defines a platform-independent infrastructure that enables the creation, storage, recognition, and exchange of metadata, and enables access to the metadata. JMI can be viewed as an extensible metadata service for the Java platform that provides a common Java programming model for accessing metadata.

BI ODBO Connector

ODBO (OLE DB for OLAP) from Microsoft has developed as an industry standard OLAP API for the Windows platform. With this connector, you can connect applications that were created with BI Java SDK with ODBO-enabled data sources such as Microsoft Analysis Services, SAS and Microsoft PivotTable Services. You can also use the BI ODBO Connector to stage data from these data sources in BI using UD Connect (see the “UD Connect” section).

The BI ODBO Connector uses ADO (ActiveX Data Objects) and ADO MD (ActiveX Data Objects Multidimensional) from Microsoft in order to support the connection to the OLAP data sources. ADO offers Access to the schema object; ADO MD also provides simple access to multidimensional data. This occurs when ADO MD enhances ADO with objects that are specifically for multidimensional data such as cubes and cell sets. With ADO and ADO MD you can search multidimensional schema, query a cube and call the results, enabling you to obtain easy access to OLAP data from the Microsoft Visual Basic, Microsoft Visual C++, and Visual J++ languages. ADO MD, like ADO uses an underlying OLE DB Provider to gain access to data. For more information, see <http://www.microsoft/library/psdk/dasdk/mdac3sc7.htm>.

BI XMLA Connector

XMLA (XML for Analysis) from Microsoft facilitates Web Service-based, platform-independent access to OLAP providers. The BI XMLA Connector enables the exchange of analytical data between a client application and a data provider over the Internet using a SOAP-based XML communication API.

With the BI XMLA Connector, you can connect applications that were created with the BI Java SDK with data sources such as Microsoft Analysis Services, Hyperion, MicroStrategy, MIS and BI. You can also use the BI XMLA Connector to stage data from these data sources in BI using UD Connect (see the “UD Connect” section). For more information, see <http://www.microsoft/library/techart/XMLAnalysis.htm>.

BI SAP Query Connector

SAP Query is a component of SAP WAS with which you can create user-specific reports without prior knowledge of ABAP programming language. The SAP Query Connector uses the SAP query so that applications that were created with the BI Java SDK can access data from operative SAP applications. You can also use the BI SAP Query Connector to stage data from these data sources in BI using UD Connect (see the “UD Connect” section).

The following table summarizes the above statements about the BI Java Connectors with regard to these factors.

- Which Connector types are available for the relational database environment and which are available for the multidimensional one
- Which technologies the individual connector types are based on
- Which data sources can be accessed respectively



	BI Java Connector	Technology based on	Provides access to
Relational	BI JDBC Connector	Sun JDBC – Standard API for RDBMS	Connectivity to over 200 JDBC drivers Examples: Teradata, Oracle, MS SQL Server, MS Access, DB2, MS Excel, Text files (such as CSV files)
	BI SAP Query Connector	SAP Query – a SAP WAS component	SAP operational applications Examples: Data in SAP OLTP systems, Ad-hoc and operational reporting
OLAP	BI ODBC Connector	Microsoft OLE DB for OLAP – implemented OLAP API for Windows, industry standard	OLE DB for OLAP-capable data sources Examples: MS Analysis Services, SAS, MS PivotTable Services
	BI XMLA Connector	Microsoft XML for Analysis – Web Service-based, platform-independent access to OLAP provider	Platform-independent access to OLAP data sources Examples: SAP BW (3.x), MS Analysis Services, Hyperion, MicroStrategy

Figure 147: Access Provided by BI Java Connectors

UD Connect

With BI, UD Connect (Universal Data Connect) provides another option for integrating data from SAP systems and external systems for analysis in BI. Both persistent and transient data staging are possible in BI.

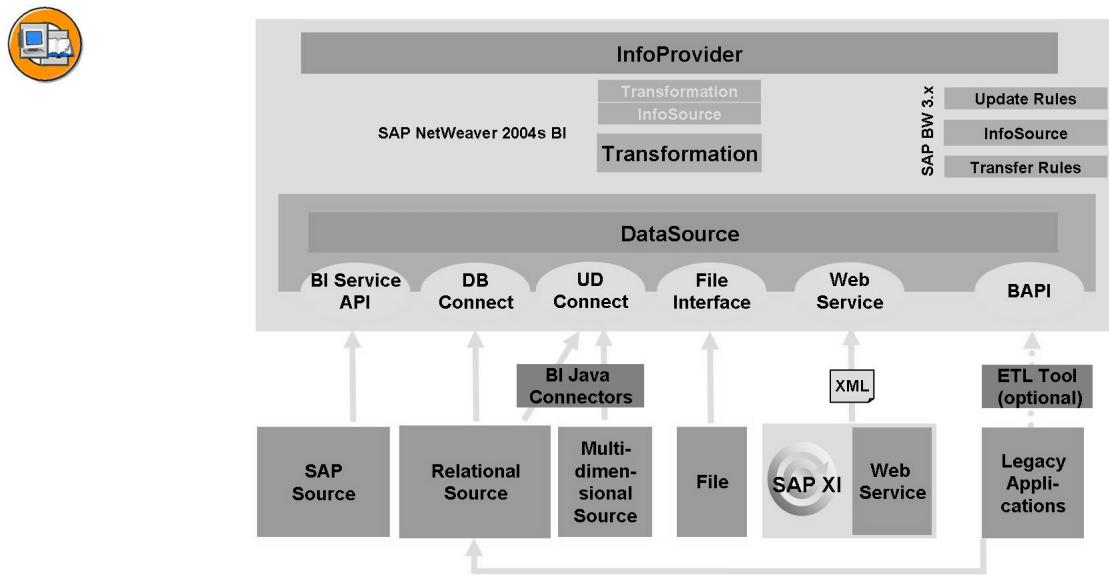


Figure 148: Data Transfer with UD Connect

UD Connect enables reporting and analysis of both SAP and non-SAP data using SAP WAS J2EE connectivity. UD Connect provides connectivity to nearly any platform or source system. For the connection to the data sources, UD Connect can use the four BI Java Connects that are available for various drivers, logs and providers:

- BI JDBC Connector
- BI ODBO Connector
- BI XMLA Connector
- BI SAP Query Connector

A prerequisite for this is that you have installed SAP WAS J2EE Engine with the BI Java components. For more information about this subject, see the SAP Service Marketplace at <http://www.service.sap.com/instguide>.

UD Connect Architecture

The following graphic illustrates the UD Connect architecture:

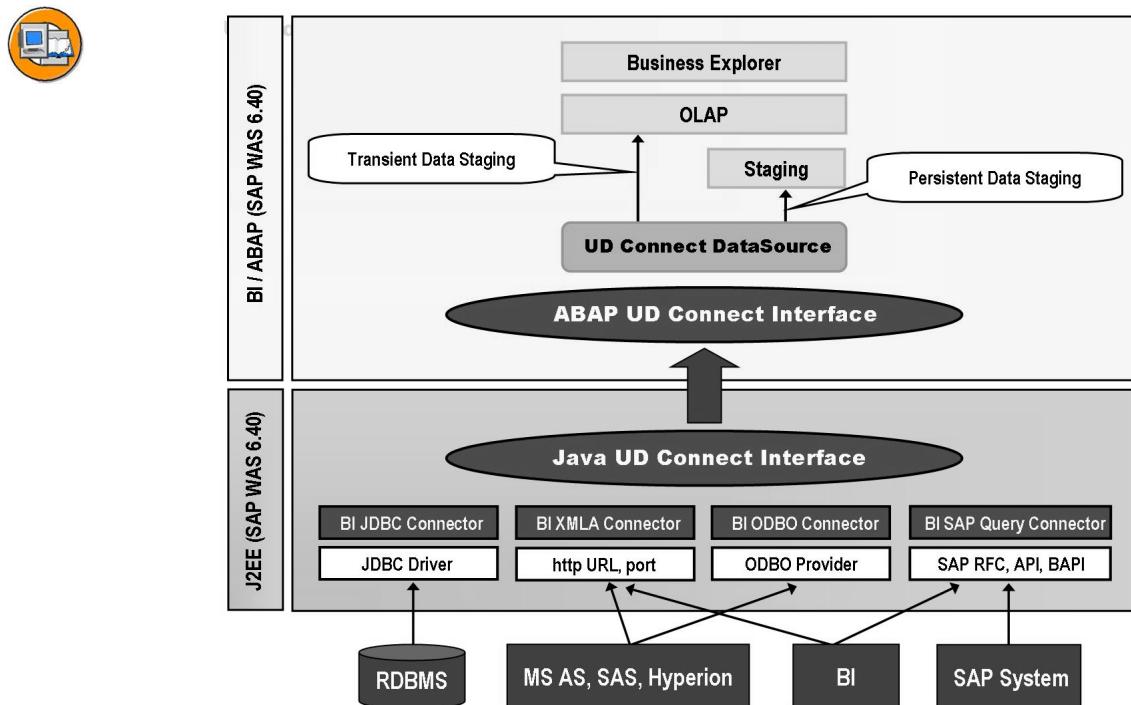


Figure 149: UD Connect Architecture

- From the UD Connect architecture you can see that UD Connect is comprised of two main components:
 - Java components on the J2EE Server of the SAP WAS
 - The ABAP component in *BI*

The Java component is responsible for communication between data sources and *BI*.
- UD Connect is displayed as a component of the J2EE Engine, together with the four BI Java Connectors that are available for various drivers, logs and providers.
- UD Connect provides Session Beans via the SAP Java Connector (JCo) that enable communication between the data sources and the *BI* server. Based on an RFC connection between the *BI* and the J2EE Engine, the Session Beans can call function modules in *BI* or can be called by the function modules.
- In *BI*, a wizard helps you to define a generic DataSource, the UD Connect DataSource. With the definition of the UD Connect DataSource, a function module (extractor) is generated with which data is read and transferred into *BI*.
- UD Connect DataSources in *BI* are then available for SAP standard scenarios to stage data in *BI*:
 - *Transient data staging*:
UD Connect DataSource → Transfer Rules → InfoSource → VirtualProvider
 - *Persistent data staging*
UD Connect DataSource → Transfer Rules → InfoSource → Update Rules → InfoProvider/DataStore Object/Characteristic InfoObject (Attributes/Texts)

UD Connect – Configuration

The configuration of the SAP WAS J2EE Engine and in *BI* required for the use of UD Connect serves on the one hand to establish the connection between the J2EE Engine and the data sources, and on the other, to establish the connection between the J2EE Engine and *BI*.

The following prerequisites must be fulfilled to this end:

- You have installed the J2EE Engine with the BI Java components.
- Using the SDM (Software Deployment Manager), you execute deployment (installation) for UDI and MMR (Metamodel Repository). When you do this, you also install the BI Java Connectors.
- If you want to access a data source using the BI JDBC Connector, you have to install the JDBC driver for the data source. For more information, see <http://www.service.sap.com/instguide> in the section:

Configuring and Checking the BI Java Connectors → Configuring the BI Java Connectors

- If you want to access a data source using the BI ODBO Connector, note that this connector can only be used with Windows 2000 / NT / XP.

For more detailed information, in *BI*, see the Installation Guide at <http://www.service.sap.com/instguide>.

1. Configuration in the J2EE Engine Administrator:

The following graphic shows the SAP J2EE Engine Administrator, in which you can configure the connection between the BI Java Connector and the data sources, as well as the connection between the SAP J2EE Engine and *BI*.

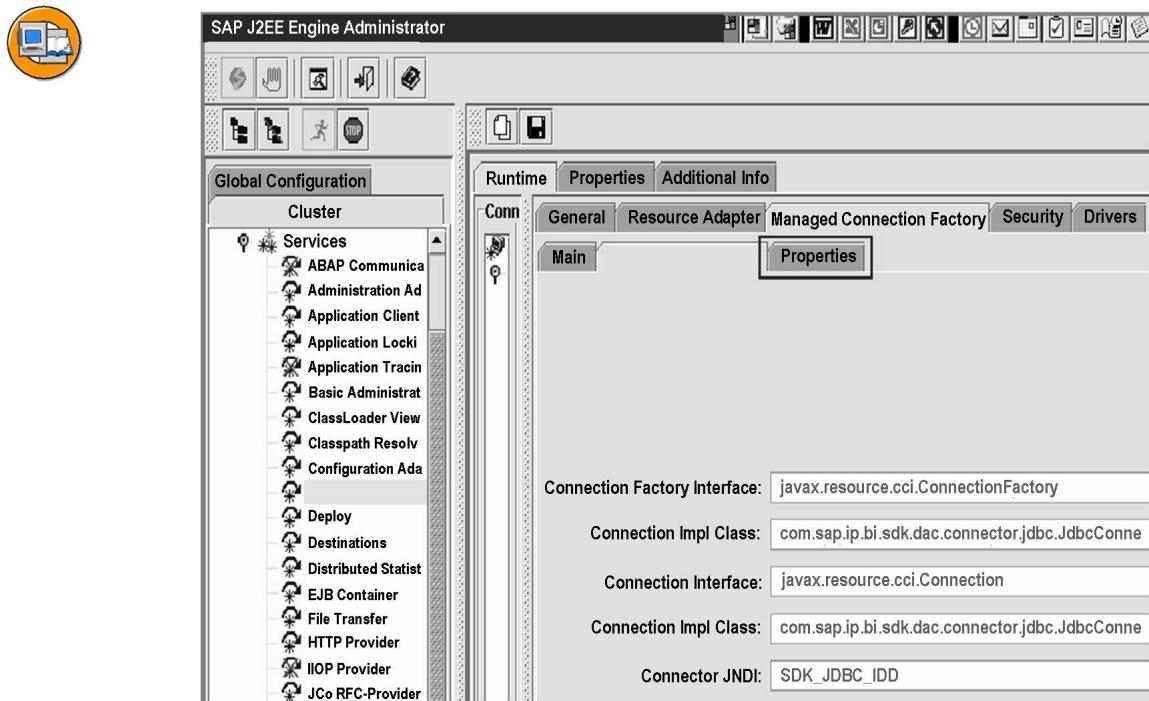


Figure 150: SAP J2EE Administrator

- *Configuration of the BI Java Connector*

In this step, you establish the connection between the data sources and the J2EE engine. Each BI Java Connector can address “n” instances. You carry out the configuration in the SAP J2EE Engine Administrator (see above graphic). To do this, on the left side of the SAP J2EE Engine Administrator screen, choose:

Cluster: Services → Node: Connector Container

Note: The Connector receives a JNDI name (Java Naming Directory Interface) in which all of the main properties are brought together. **SDK_** is assigned as the prefix for name of the Connector so that the BI Java Connector is recognized by UD Connect. The definition of the UD Connect DataSource references this name later. For more information, see <http://www.service.sap.com/instguide> in the section *Configuring and Checking the BI Java Connectors → Configuring the BI Java Connectors*

- *Setting up the RFC destination for BI:*

- To do this, on the left side of the SAP J2EE Engine Administrator screen, choose:

Cluster: Services → Node: JCo RFC Provider

- Enter data for the following designations in the *Runtime* tab page:

Under RFC destination:

Program ID, gateway host, gateway service, number of processes
under Repository:

Application Server Host, System Number, Client, Language, User Name, Password



Hint: The program ID entered here must correspond to the program ID that you enter when maintaining the RFC destination in BI.

- Choose *Set*.

The RFC destination is displayed on the left side of the *Runtime* tab page and is checked implicitly. The symbol in front of the name of the RFC destination displays whether it is executable. The repository is first tested at runtime.

2. Configuration in BI:

Setting up the RFC destination for SAP J2EE Engine:

You set up the RFC destination for the J2EE engine in *RFC Destination Maintenance* in *BI* in the transaction *SM59*.

- a) Choose *Edit* → *Create*.
- b) Enter a name and description for the destination.
- c) Choose the connection type *T* (Start an external program using TCP/IP).
- d) On the *Technical Settings* tab page, maintain the following parameters:

UD Connect – RFC-Destination in BI

Under activation type:

Registered server program

Under registered server program:

Program ID



Hint: The program ID entered here must correspond to the program ID that you enter when maintaining the J2EE Engine in BI.

Under gateway options:

Gateway host, gateway service

- e) Save and test the connection.

With the configurations described above, you have established the connection between data sources and the J2EE Engine and between the J2EE Engine and the *BI*. In *BI*, you can now define a UD Connect DataSources for an InfoSource for transient/persistent data staging. The procedure for this is described below.

Definition of UD Connect DataSources

Before you can transfer the data from UD Connect sources to *BI*, you have to communicate the metadata for the data to be extracted to *BI* using DataSources.

The name below is used for defining a UD Connect DataSource.

- **UD Connect Source**

UD Connect Sources are instances that can be addressed as data sources via the BI Java Connectors. Data sources can be relational data sources, OLAP-compliant (multidimensional) data sources, XMLA-compliant (multidimensional) data sources and SAP applications.

- **UD Connect Source Object**

Relational and multidimensional data stores in the data source are designated as UD Connect source object (such as tables and cubes).

- **Source Object Element**

Source object elements are the components of UD Connect source objects, such as fields for tables, characteristics and key figures of cubes.

Define a UD Connect DataSource

The definition of a UD Connect DataSource is described in the procedure at the end of this lesson.

Use the menu:

Data Warehousing Workbench: Modeling → Source Systems → UD Connect → Specific UD Connect Source System → Create DataSource

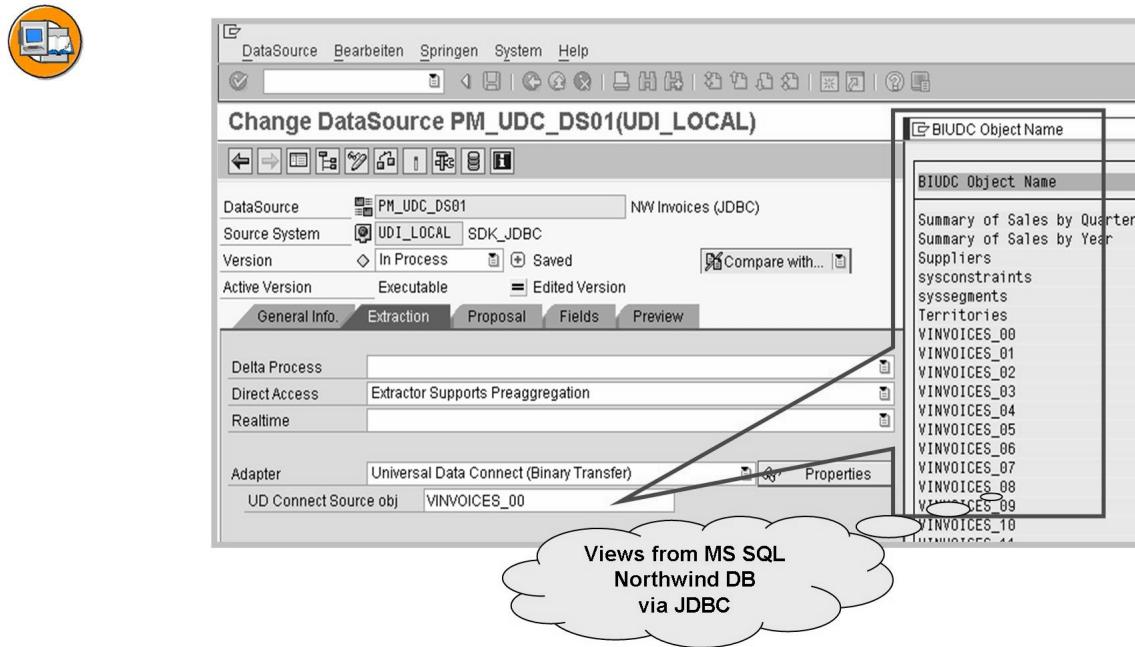


Figure 151: Defining UD Connect DataSource I

The dialog window opens. Here, you can make settings to assign a UD Connect source object to a DataSource and then generate the DataSource. On the *Extraction* tab, select the appropriate UDC object in the field *UD Connect Source obj..*

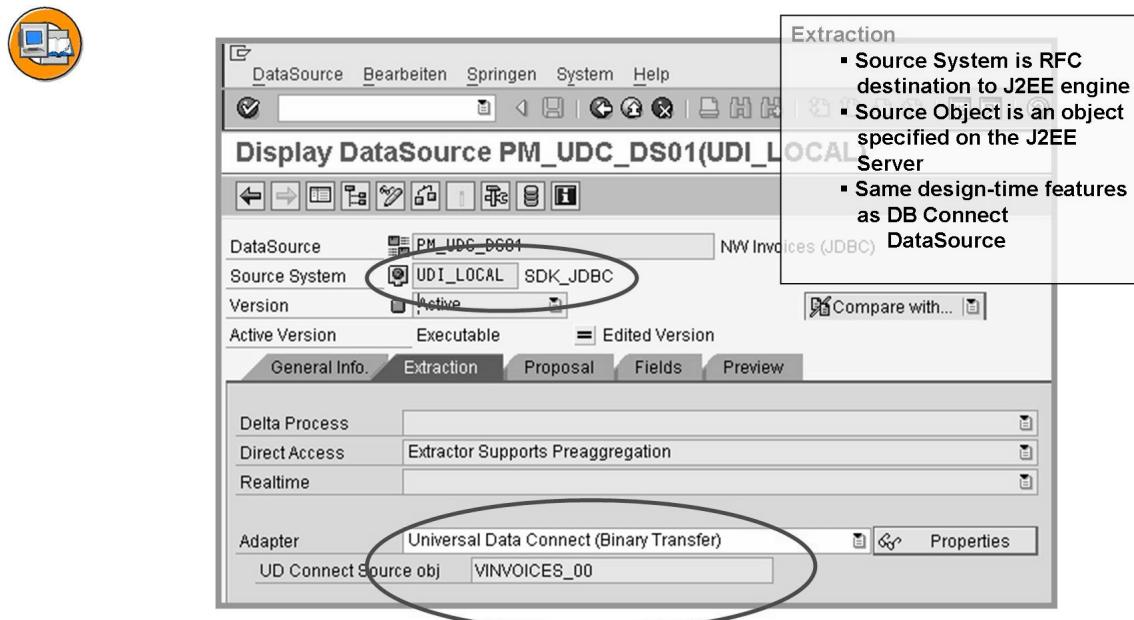


Figure 152: Defining UD Connect DataSource II

You have created a DataSource for the transient or persistent data staging with UD Connect. Furthermore, you can display the metadata for this DataSource (for example, extraction structure, extractor) in the table *RSOLTPSOURCE* in the *BI* system.

For UD Connect DataSource details, see the Procedure at the end of this lesson.

BI Java SDK

With the BI Java Software Development Kit (SDK), you can create analytical applications to access, edit and display both multi-dimensional and relational data. The target audience is Java developers that have experience with business intelligence.

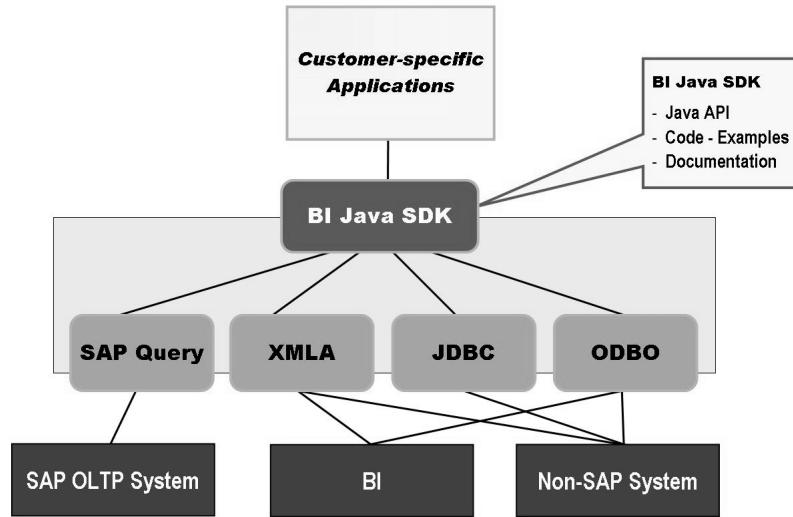


Figure 153: BI Java SDK

The SDK distribution package provides a complete documentation set. It contains a developer's guide with step-by-step tutorials, JavaDocs including package and overview documentation and HTML navigation that links the SDK distribution package and the documentation to one another.

SDK examples provide easy-to-use Java servlets that illustrate many aspects of BI query APIs, as well as step-by-step connection options.



Figure 154: BI Java SDK: Documentation for Developers

Although the main objective of the BI Java SDK is to simplify programming compared with the *BI* OLAP Engine in a Java environment, the APIs can also be used to access non-*BI* and even non-OLAP data sources, such as relational JDBC data sources. In this way, programs can work with one, universal approach during the entire development of an application.



Creating DataSources for UD Connect

1. Select the application components in which the DataSource is to be created and choose *Create DataSource*.
2. In the next screen, enter a technical name for the DataSource, select the type of the DataSource and choose *Copy*. The DataSource maintenance screen appears.
3. Go to the *General* tab page.
 - a) Enter descriptions for the DataSource (short, medium, long).
 - b) If necessary, specify if the DataSource builds an initial non-cumulative and supplies potentially duplicate data records within a request
4. Go to the *Extraction* tab page.
 - a) Define the delta method for the DataSource.
 - b) Specify whether you want the DataSource to support direct access to data.
 - c) UD Connect does not support real-time data acquisition.
 - d) The system displays Universal Data Connect (Binary Transfer) as the adapter for the DataSource. Choose Properties if you want to display the general adapter properties.
 - e) Select the UD Connect source object. The connection to the UD Connect source is established. All source objects that are available in the selected UD Connect source can be selected using input help. Source objects can be multi-dimensional storage or relational objects.
5. Go to the *Proposal* tab page. The system displays the elements of the source object (for JDBC for example fields, or for XMLA and ODBO characteristics and key figures) and creates a mapping proposal for DataSource fields. The mapping proposal is based on the similarity of the names of the source object element and DataSource fields and the compatibility of the respective data types. Note that source object elements can have a maximum of 90 characters. Both upper and lower case are supported.
 - a) Check the mapping and change the proposed mapping as required. Assign the non-assigned source object elements to free DataSource fields. You cannot map elements to fields if the types are incompatible. The system produces an error message in this case.
 - b) Under *Copy to Field List*, select the fields that you want to transfer to the field list for the DataSource. All fields are selected by default.

Continued on next page

- c) If required, change the values for the key fields of the source. These fields are generated as a secondary index in the PSA. This is important in ensuring good performance for data transfer process selections, in particular with semantic grouping.
- d) If required, change the data type for a field.
- e) Specify whether the source provides the data in the internal or external format.
- f) If you choose an External Format, ensure that the output length of the field (external length) is correct. Change the entries, as required.
- g) If required, specify a conversion routine that converts data from an external format into an internal format.
- h) Select the fields for which you want to be able to set selection criteria when you schedule a data request using an InfoPackage. Data for this type of field is transferred in accordance with the selection criteria specified in the InfoPackage.
- i) Choose the selection options (such as EQ, BT) that you want to be available for selection in the InfoPackage.
- j) Under *Field Type*, specify whether the data to be selected is language-dependent or time-dependent, as required.



Note: If you did not transfer the field list from a proposal, you can define the fields of the DataSource directly. Choose *Insert Row* and enter a field name. You can specify InfoObjects in order to define the DataSource fields. Under *Template InfoObject*, specify InfoObjects for the fields of the DataSource. This allows you to transfer the technical properties of the InfoObjects into the DataSource field. Entering InfoObjects here does not equate to assigning them to DataSource fields. Assignments are 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.

- 6. Maintain the *Fields* tab page. Here you edit the fields that you transferred to the field list of the DataSource from the *Proposal* tab page.
 - a) Under *Transfer*, specify the decision-relevant DataSource fields that should be available for extraction and transferred to BI.
- 7. Check, save and activate the DataSource.

Continued on next page

8. Go to the *Preview* tab page. If you select *Read Preview Data*, the number of data records you specified in your field selection is displayed in a preview. This function allows you to check whether the data formats and data are correct.

Result

The DataSource is created and added to the DataSource overview for the UD Connect source system in the application component in the Data Warehousing Workbench. When you activate the DataSource, the system generates a PSA table and a transfer program.

You can now create an InfoPackage. You define the selections for the data request in the InfoPackage. The data can be loaded into the entry layer of the BI system, the PSA. Alternatively you can access the data directly if the DataSource allows direct access and you have a VirtualProvider in the definition of the data flow.



Lesson Summary

You should now be able to:

- Describe the concept and architecture of UDI
- Name the key components of UDI
- Get an overview of the functions of the BI Java Connectors
- Outline the UD Connect architecture
- Explain the significance of the BI Java SDK



Unit Summary

You should now be able to:

- Describe the concept and architecture of UDI
- Name the key components of UDI
- Get an overview of the functions of the BI Java Connectors
- Outline the UD Connect architecture
- Explain the significance of the BI Java SDK

Unit 8

XML-Based Data Acquisition

Unit Overview

This unit introduces data extraction via XML (extensible markup language). XML has proven to be a reliable, stable standard for the transfer of data. The current business environment demands open data interchange with your customers, subsidiaries and other business partners. Frequently, different data formats are used. To ensure unified, stable data extraction, you can extract this data in XML format.



Unit Objectives

After completing this unit, you will be able to:

- Explain the underlying standards for XML-based extraction
- Name the necessary objects in the BI system
- Implement the common framework for bringing together the different extraction mechanisms
- Describe the function of the SOAP service DataSource 3.x in the SAP Web Application Server
- Explain how the SOAP service DataSource 3.x enables data acquisition from an application
- Describe the important business benefits provided by Web services
- Discuss the significance of Web service standards
- Describe how to implement BI Web services for XML data load
- Gain an overview of the structure and terminology of an SAP PI implementation
- Give examples for SAP PI and BI integration scenarios
- Describe the technologies with which BI and SAP PI can work together
- Describe the implementation of an SAP PI / BI connection

Unit Contents

Lesson: Introduction to XML-Based Extraction	393
Lesson: Using the Web AS SOAP Service (Optional)	401
Lesson: XML Data Acquisition Using a Web Service	408

Exercise 12: Web Service DataSource.....	417
Lesson: XML Data Acquisition Using SAP PI.....	427

Lesson: Introduction to XML-Based Extraction

Lesson Overview



Lesson Objectives

After completing this lesson, you will be able to:

- Explain the underlying standards for XML-based extraction
- Name the necessary objects in the BI system
- Implement the common framework for bringing together the different extraction mechanisms

Business Example

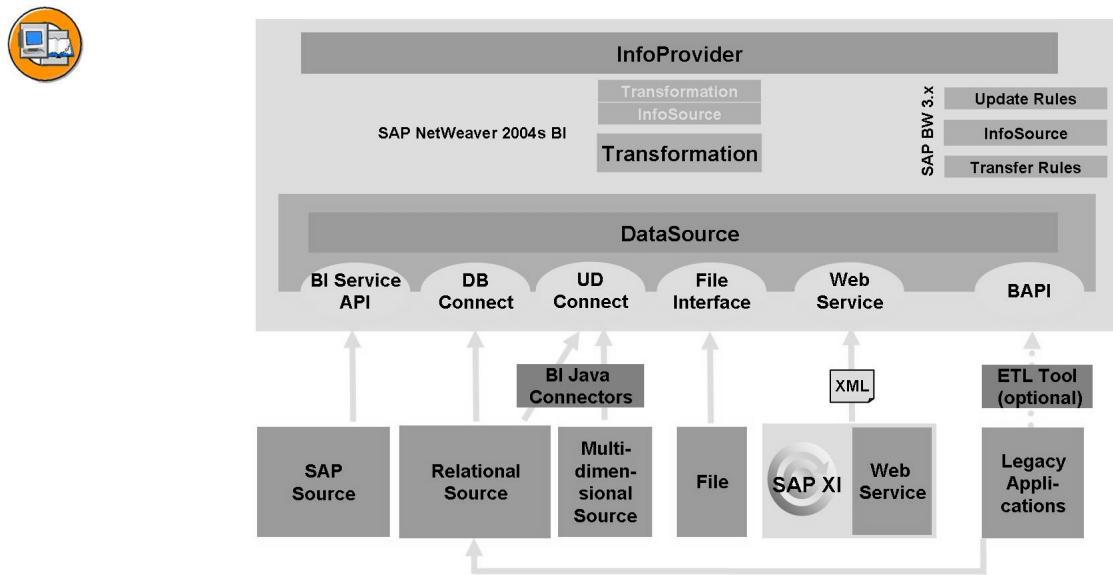
The current business environment demands a more open data interchange with your customers, subsidiaries and other business partners.

There is an increasing need to incorporate data of various formats and source systems. XML has proven to be a reliable, stable standard for the transfer of data.

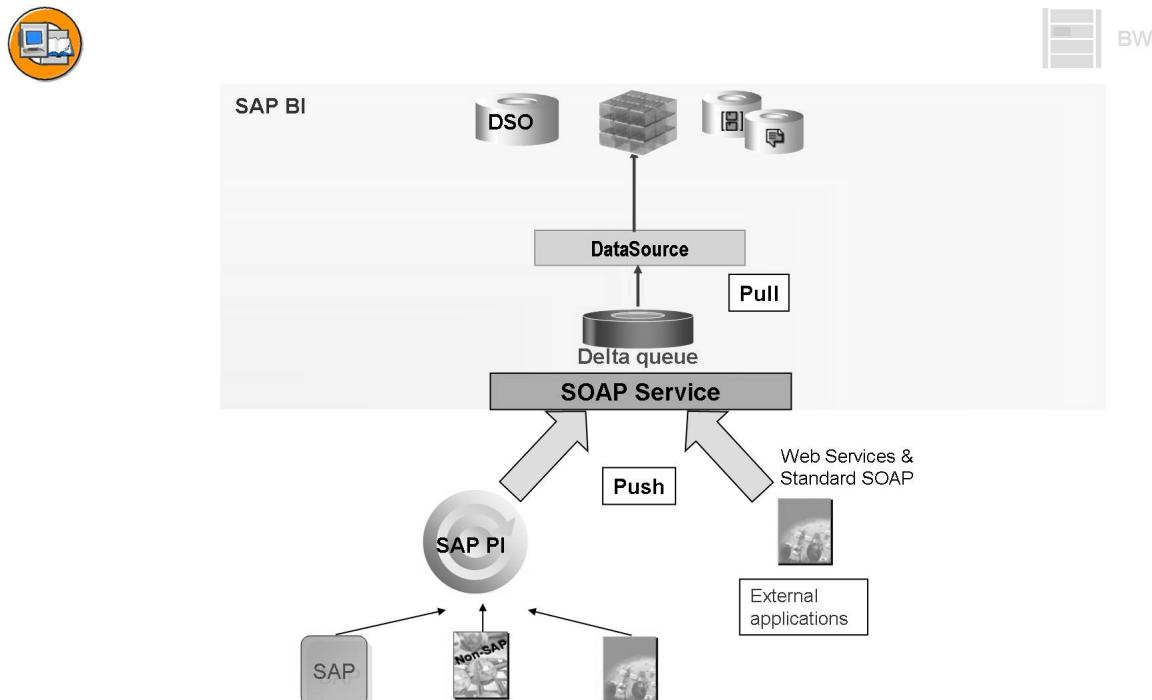
In contrast to the general “Pull” extraction strategy, the transfer of data is to be initiated by the source systems (“Push” mechanism).

Introduction to XML-Based Extraction

Data is generally transferred into BI by means of a data request, which is sent from BI to the source system (pull from the scheduler). You can also send the data specifically to the BI system using Web services. This is a data push into the BI system. You use Web services for real-time data acquisition. The data is first written to the PSA of the BI system. From there, the data is controlled by a background process, or daemon, which runs at regular intervals; the data is updated to a DataStore object and is then immediately available for operational reporting. In SAP NetWeaver 2004s, you generate Web services for data loading when you activate a DataSource defined in the BI system. The Web services provide you with WSDL descriptions, which can be used to send data to BI regardless of the technology used. The BI Data Staging architecture is illustrated below:

**Figure 155: BI Data Staging: Architecture**

As a rule, data transfer in BI takes place by means of a data request that is sent from BI to the source system (“Pull” from the scheduler). You can also send the data to SAP BI by external control. This is a data “Push” to the SAP BI system.

**Figure 156: XML-Based Extraction: Overview**

The data is “Pushed” into the delta queue. It is then transferred to potential data targets using the standard “Pull” mechanism for the BI system.

A data push is possible for a number of scenarios:

- Data transfer using the SOAP service SAP Web AS
- Data transfer using a web service
- Data transfer using SAP XI

In all three scenarios, data transfer takes place using XML-based transfer mechanisms that meet Simple Object Access Protocol (SOAP) requirements.

Web Application Server (Web AS) and open standards

The SAP Web Application Server uses the basic open standards to support the transfer of data in XML format.

- eXtensible Markup Language (XML)
- Simple Object Access Protocol (SOAP)

Open integration can only be achieved if it is based on generally accepted standards.

Basic web service standards are implemented in the SAP Web Application Server:

- **eXtensible Markup Language (XML)**

XML is a simple and extremely flexible text format. Originally designed to meet the challenges of large-scale electronic publishing, XML also plays an increasingly important role in exchanging a wide variety of data on the Internet and elsewhere.

- **Simple Object Access Protocol (SOAP)**

SOAP is a simply designed protocol for exchanging information in a decentralized, distributed environment. The format is XML-based. A sample is shown below:



```
<?xml version= "1.0" ?>
- <DATASOURCE>6AFILEDS99
</DATASOURCE>
XML is...
- <DATA>
  <item>
    <VENDOR>OTTO</VENDOR>
    <MATERIAL>Soap</MATERIAL>
    <DATE>20010104</DATE>
    <UNIT>KG</UNIT>
    <AMOUNT>1234567</AMOUNT>
  </item>
  <item>
    <VENDOR>FRITZ</VENDOR>
    <MATERIAL>Aftershave</MATERIAL>
    <DATE>20010213</DATE>
    <UNIT>ml</UNIT>
    <AMOUNT>100</AMOUNT>
  </item>
</DATA>
```



the universal format for structured information on the Web

Figure 157: XML - eXtensible Markup Language

XML can be used on the Web and is a markup language that is:

- Simple
- Text format
- Vendor-independent/standardized
- Generally accepted
- User-extensible
- Suitable for complex structures
- Suitable for validity checks

The SAP BI system is currently not capable of processing hierarchical XML structures.

XML Schema - Document Types

XML Schema

- is a means for defining the
 - Structure (“This element contains the following elements that contain ...”)
 - Data type (“This element contains an integer.”)
 - and constraints (“Value range from 0 to 999, maximum length 3.”)
- of XML documents
- is designed for reuse and extensibility
- allows validation of XML instances
- documents (XSDL) are formulated in XML
- is used in the Interface Repository, the Web Services Infrastructure and Exchange Infrastructure

You can describe the structure of a schema with an *XML Schema*. The *XML Schema* also describes the particular structure of an XML document (elements, attributes, hierarchy). This language allows you to define simple and complex data types. The XML schema definition for the data transferred to the BI system is derived from the definition of the file or XML DataSource.

This type of document is used to communicate the various extraction mechanisms described in this unit

- XML: Data description
- XSDL: Description of structure data & services

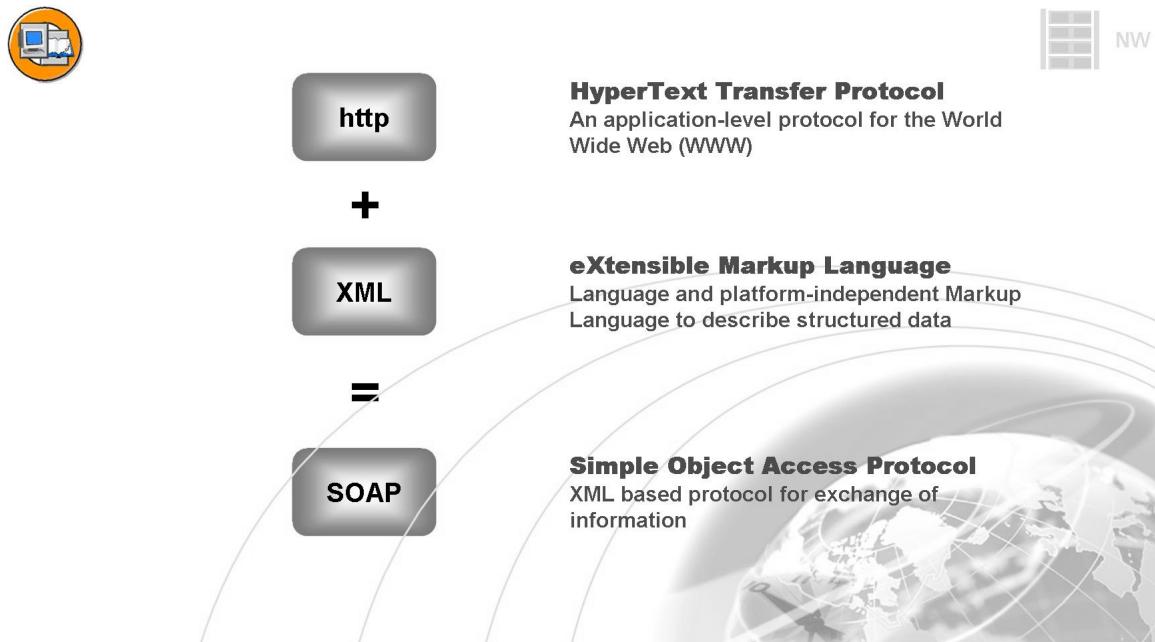


Figure 158: SOAP: Simple Object Access Protocol

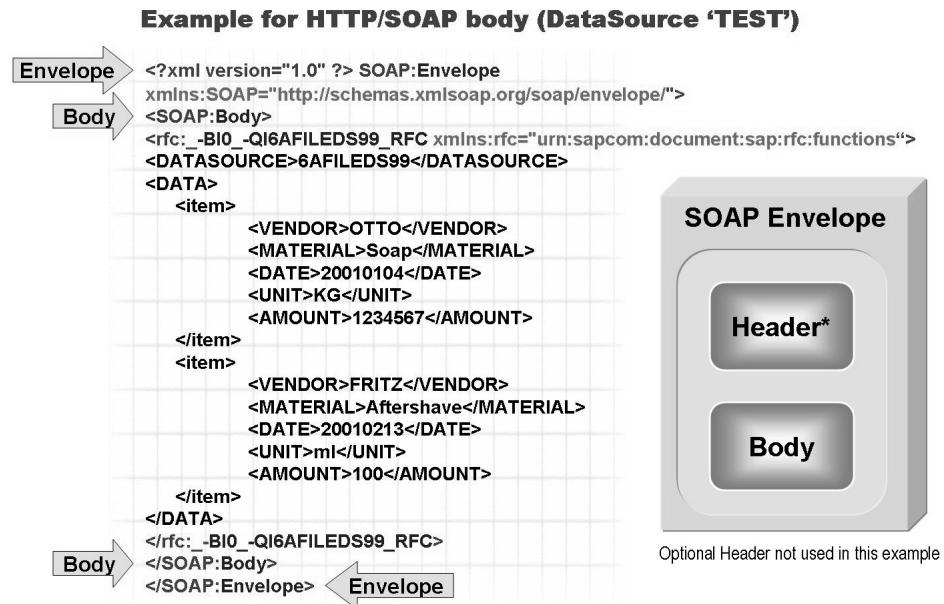


Figure 159: SOAP Body for HTTP POST request

SOAP messages consist of an **Envelope**, an optional **Header** and a **Body**.

The **Envelope** must be the first element in a SOAP message. It serves to determine that an XML document is a SOAP message and encapsulates all the other parts of a message. The Envelope contains information about the functions and services in a message and the processing method for this message.

The **Header** (not used in the above example) contains additional, platform-independent syntax such as authorization details. The Header must follow the Envelope tag.

The **Body** is where the application-specific data is located. Within the body, the DataSource, the corresponding RFC enable function module, and the data to be transferred are defined.

In the above example the message calls a BI function module.

Note: Data cannot be arranged in a hierarchical structure; it must be flat. In other words, document data must not include detailed information at header and item level.

Source data in XML format can be transferred to SAP BI over the Internet and written to the delta queue.

A file DataSource is used to load the historical data for the first time. Further data loads populate the delta queue directly.

In this scenario, data is loaded to the delta queue with SOAP. A DataSource with a SOAP connection is created. This DataSource has a corresponding RFC function module that pushes the data to the delta queue.



Lesson Summary

You should now be able to:

- Explain the underlying standards for XML-based extraction
- Name the necessary objects in the BI system
- Implement the common framework for bringing together the different extraction mechanisms

Lesson: Using the Web AS SOAP Service (Optional)

Lesson Overview

This lesson describes how to use the SOAP service DataSource 3.x in the SAP Web Application Server for loading data and how to address this service from an application.



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the function of the SOAP service DataSource 3.x in the SAP Web Application Server
- Explain how the SOAP service DataSource 3.x enables data acquisition from an application

Business Example

A company has set up a Internet shop for its customers. The customers can look through the catalogs online and order the product via the Internet.

As soon as an order is placed, the information is transferred to the SAP BI system in order to update the stock listing.

Using the SOAP Service for Loading XML Data

You can use a SOAP service DataSource 3.x in the Web Application Server to load XML data.

SOAP-based data transfer is still supported in data models with 3.x objects. Real-time data acquisition, however, is not possible for these models.



XML Data Load Using the Web AS SOAP Service DataSource 3.x

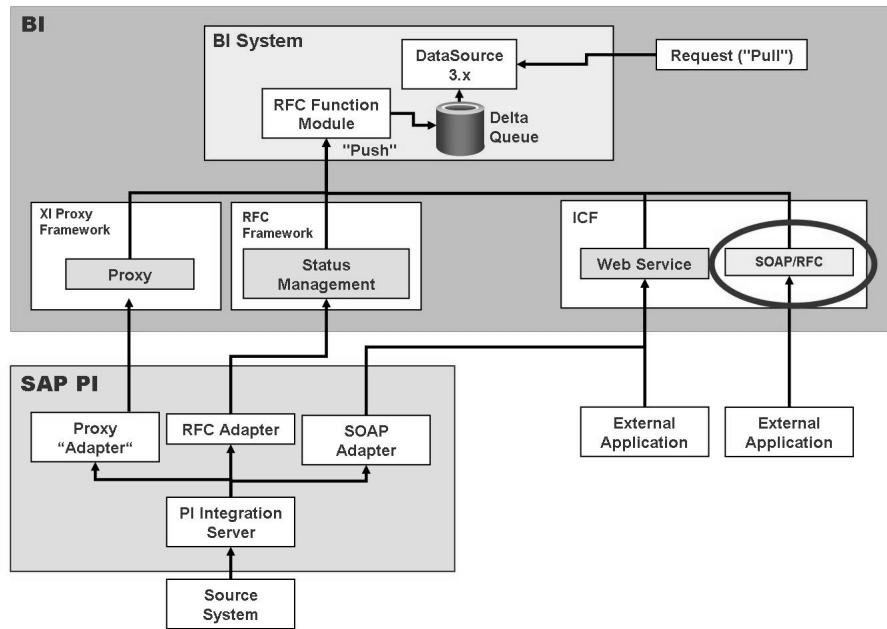


Figure 160: XML Data Load Using the Web AS SOAP Service DataSource 3.x

BI uses a SOAP service DataSource 3.x with the SAP Web Application Server. By using this service you can transfer XML data that is sufficient for the SOAP protocol to RFC-enabled function modules in the ABAP environment. On the basis of RFC compatibility, the function module can be accessed automatically using one of the assigned HTTP handlers from SAP for supporting SOAP. The SOAP service checks the XML data for correct syntax and converts it to ABAP fields. In doing so, the XML data has to be assigned according to an XML schema definition, which is derived from the definition of a flat file or XML DataSource 3.x. Data is transferred in BI with a push to the delta queue of the generated DataSource 3.x.

The application logic (writing to the delta queue) is provided as an ABAP RFC function module.

The function module is generated when the DataSource 3.x is created with *SOAP connection*.

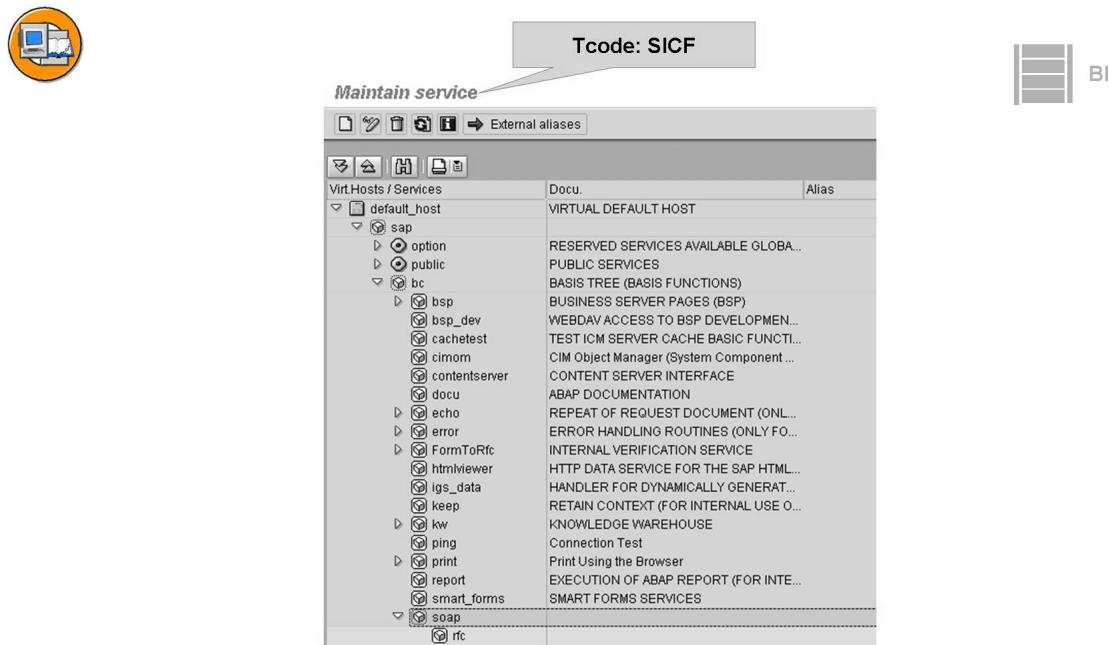


Figure 161: Internet Communication Services in BI

Use transaction SICF (System Internet Communication Framework) to see the maintenance settings for this service.

You can find the service to be called under `/sap/bc/soap/rfc`.

The relevant HTTP port for calling the Web AS SOAP service is listed here. You can check whether the SOAP service is active. To do this, choose *Go to → ICM Monitor*. Then choose *Go to → Services*. The port to be used and the status of the service are displayed in the table for the HTTP system log.

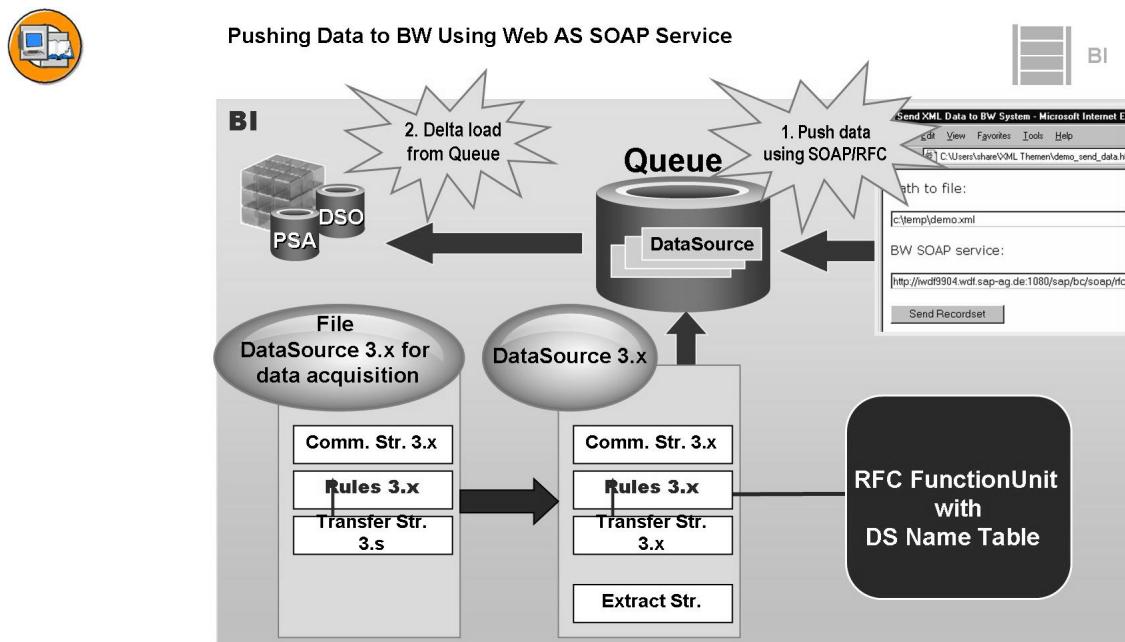


Figure 162: Pushing the Data to BI via Web AS SOAP Service

This figure shows the steps for pushing data from an application to the BI system using the Web AS SOAP service.

1. Start the SOAP RFC service to push data into the delta queue in BI.
2. Send the data to be loaded in XML format using the HTTP port specified under `/sap/bc/soap/rfc` to the SAP Web Application Server SOAP service.
3. Load data from the delta queue to BI InfoProviders.

The following figures explain these steps in more detail.

Data is pushed into the queue by taking the XML format and parsing it into the DDIC structure of the called RFC function module.

The data is then available in the delta queue of the BI system and can be transferred (pulled) from there using a normal loading process to other InfoProviders.

→ **Note:** Creating the push event is a manual process (for example, saving a shopping basket in a web application). This method is not suitable for mass data loading processes.

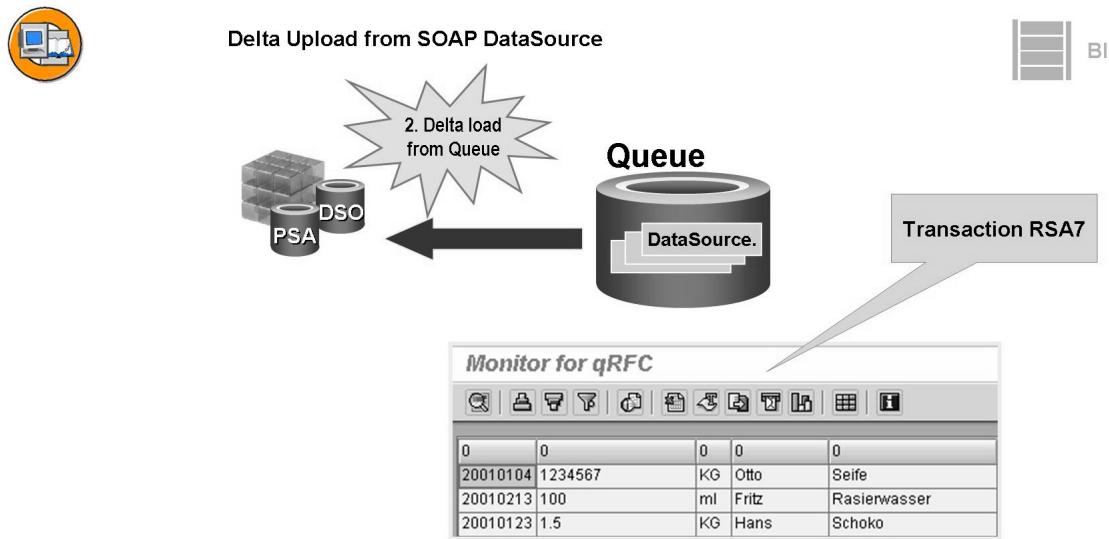


Figure 163: Delta Upload from SOAP DataSource 3.x

After the data has been pushed to the delta queue, it can be viewed in the transaction RSA7. To display the data for a specific DataSource 3.x , mark the DataSource 3.x and choose *Display Data*.

The data from the delta queue can be passed on to the InfoProviders in the BI system by scheduling a delta InfoPackage for the BI DataSource 3.x with *SOAP connection*. The following steps are involved in passing data to InfoProviders:

- Maintenance of XML data type definition via file-based DataSource 3.x definition
- Automatic generation of a BI DataSource 3.x and an RFC-enabled function module to build the interface for the delta queue using HTTP/SOAP

The file-based DataSource 3.x has the following limitations:

- The file-based DataSource 3.x must be assigned to “flexible” type InfoSources.
- Transfer rules must be active so that generated data elements can be reused.

The generated BI DataSource 3.x has the following properties

- DataSource 3.x for transaction data (this means flexible staging)
- Data extraction in delta mode only
- Delta queue activated by initialization without data loading
- Selection fields and delta process copied from the file-based DataSource 3.x

The RFC-enabled function module has the following properties:

Function group name: /BI0/QI<DataSource 3.x>

Function module name: /BI0/QI<DataSource 3.x>_RFC

- Import parameter “datasource”
- Tables parameter “data”

Reliability Concerns

Synchronous communication means that you can guarantee that the data is transferred.

Dedicated error messages are sent back to the client. The client in turn can decide to resend a given data packet to make sure that it was transferred successfully by the service.

It is currently not possible to guarantee that data is transferred only once. This is because transaction IDs are not analyzed to prevent a data package that has already been transferred from being posted.

With the ability of DS objects to remain consistent even if data is updated more than once (delta method for after images “AIM”), it is nevertheless possible to guarantee correct data in the most important cases.



Lesson Summary

You should now be able to:

- Describe the function of the SOAP service DataSource 3.x in the SAP Web Application Server
- Explain how the SOAP service DataSource 3.x enables data acquisition from an application

Lesson: XML Data Acquisition Using a Web Service

Lesson Overview

This lesson discusses the benefits of using a Web service to load XML data. It also describes how to implement BI Web services for XML data load.



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the important business benefits provided by Web services
- Discuss the significance of Web service standards
- Describe how to implement BI Web services for XML data load

Business Example

Your organization has to send transaction data to BI from a non-SAP system. After much analysis it was decided that the most cost-effective solution would be to load the data to BI directly over the Internet. The functions provided by BI Web services for XML data load offer a viable solution to this task.

Using Web Services to Load XML Data

Data is generally transferred into BI by means of a data request, which is sent from BI to the source system (pull from the scheduler). You can also send the data specifically to the BI system using Web services. This is a data push into the BI system.

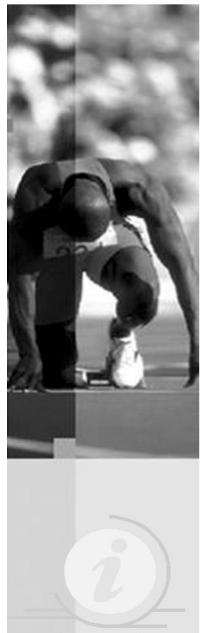
You use Web services for **real-time data acquisition**. The data is first written to the PSA of the BI system. From there, the data is controlled by a background process, or daemon, which runs at regular intervals; the data is updated to a DataStore object and is then immediately available for operational reporting.

In SAP NetWeaver 2004s, you generate Web services for data loading when you activate a DataSource defined in the BI system. The Web services provide you with WSDL descriptions, which can be used to send data to BI regardless of the technology used.

Web Services are preferable to the SOAP-based XML transfer of data for the following reasons:

- The BI server SOAP interface can ensure guaranteed delivery, since an XML message is returned to the client upon success as well as failure. If the client receives an error or no message at all (due to connection termination when sending a success message, for example), the client can resend the data.
- It is not currently possible to ensure guaranteed delivery only once, since there is no match on the transaction-ID level using which it is possible to determine that a data package was 'inadvertently' resent and is not to be updated. If the deltas are built with after-images (delta process AIM), the update to a DataStore object can, however, consistently deal with data sent excessively, as long as serialization is guaranteed. The serialization is the task of the client.

Web services are independent and self-describing application functions that are processed by open Internet standards. Web services can be used to load XML data to external systems.



- **WEB SERVICES are**
- **INDEPENDENT**
- **and SELF-DESCRIBING**
- **APPLICATION FUNCTIONS**
- **that are**
- **PROCESSED**
- **by**
- **OPEN INTERNET STANDARDS**

Figure 164: Web Services Definition

SOAP is the SAP technology for connecting SAP Exchange Infrastructure (SAP PI) and SAP NetWeaver Business Intelligence (BI).

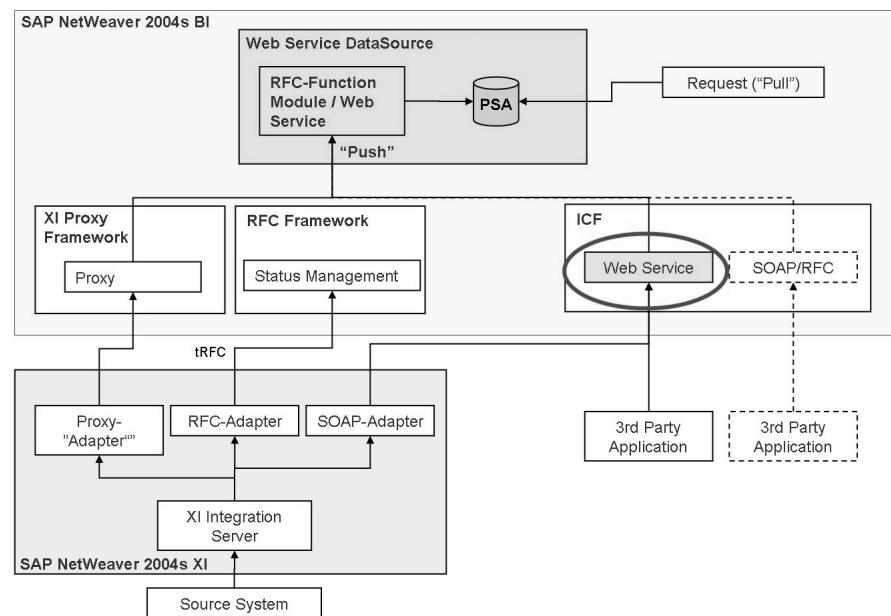


Figure 165: XML Data Acquisition Architecture using a BI Web Service

SAP already has many interfaces that can be used as a basis for Web services. Existing **application functions** are therefore used and offered as Web services, through simple **configuration** without additional coding. There is a unified concept for Java and ABAP, which means it is possible to transform Java classes, BAPIs, IDocs, function modules, and so on to a Web service.

Web services are based on and use **open standards**.

The following figure lists examples of Web services.



- Intelligent product catalog search
- Product availability check
- Pricing inquiry
- Customer credit check
- Order status check
- Vendor managed inventory
 - Demand forecasts, stock replenishment ...
- Dynamic auctioning and bidding
- Publishing and analyzing financial reports (XBRL based)
- Electronic bill presentment and payment
- Matching vacancies and job applicant profiles
- Postal service address check
- UDDI registration and discovery services
- Automated web searches (Google)



Figure 166: Web Services: Examples

XML-based business reporting language (XBRL) provides the formats that are used to exchange the financial data from balance sheets and other financial statements.

For accounting services, financial analytics, market research and so on, there are new options for collecting and analyzing information.

New options also exist for electronic bill presentment and payment. For example, banks offer online access to bills, provide easier payment methods to customers, and relieve companies of laborious collection processes.

In human resources, job offerings and job searching are supported by HR-XML, a format that enables automatic matching of vacancies with job offerings.

A simple, but very useful Web service is address and zip code checking.

Google, a popular Internet search engine, provides an application programming interface (API) for program-based Web searches.

The following figure shows the basic Web services framework.

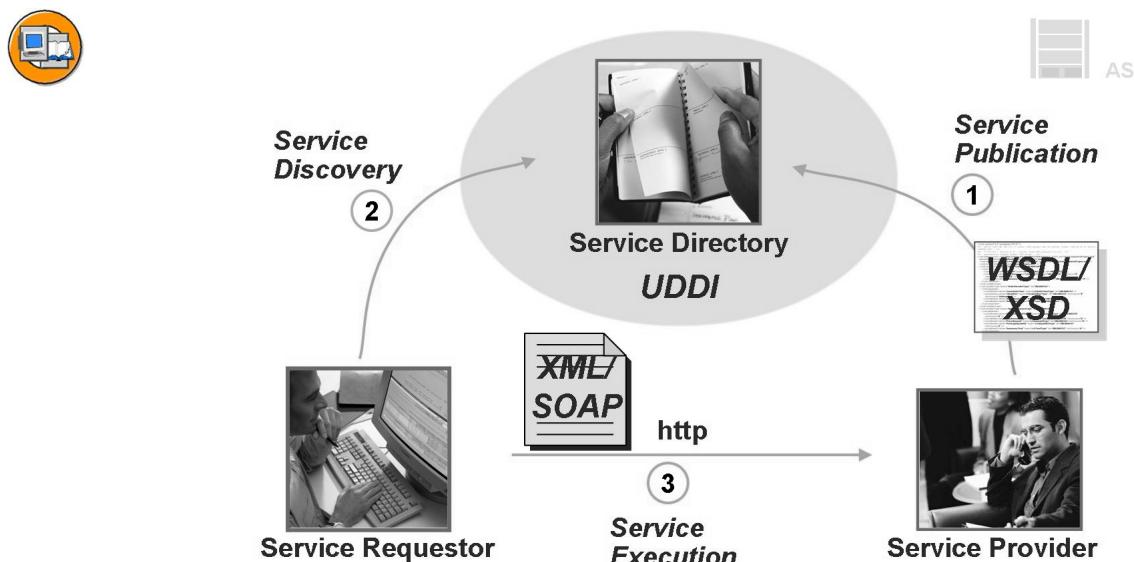


Figure 167: Web Service Paradigm

A few of the elements in this framework are described in the following text:

Service provider

Within the SAP Web AS, the inside-out approach is used to develop functions that can be made available as a Web service. After implementing the functions, a Web service interface has to be created that is visible to the Web service user. This interface provides an abstraction layer and consequently, independence from the specific implementation. The Web service is configured on the basis of this interface. You can access runtime on it. Full UDDI client capabilities can be used to publish the Web services to a UDDI registry.

Service directory

Web service definitions and deployed Web services can be stored in a UDDI registry. WSDL documents provide the basis for the Web service client and can be found in the UDDI using a browser or the standard UDDI APIs. The SAP Web AS provides UDDI client server functions. You can search in and publish to all registries that conform to the standard. A UDDI server with the complete range of functions is shipped as part of the SAP Web AS, so customers can create their own registries. SAP also offers a public UDDI Business Registry under uddi.sap.com.

Service Requestor

The SAP Web AS allows you to integrate Web services using the WSDL file as input for generating a Web service client.

The following figure lists the open standards used by Web services.



The SAP Web Application Server implements the basic Web Services standards.

- Web Service Description Language (WSDL)**
- Universal Description, Discovery and Integration (UDDI)**
- eXtensible Markup Language (XML)**
- SOAP**

Figure 168: Web Services and Open Standards

Open integration can only be achieved if it is based on generally accepted standards.

Basic Web service standards are implemented in the SAP Web Application Server:

- Extensible Markup Language (XML)
- SOAP
- Web Service Definition Language (WSDL)
- Universal Description, Discovery and Integration (UDDI)

The following figure lists the standards that apply to use of the **Web Service Definition Language** (WSDL).



WSDL describes the basic characteristics of a Web service.



WSDL is an XML format describing web services that are available over the Internet.

A WSDL document determines:

- The functions executed by a web service**
- The location of a web service**
- How a web service can be accessed**
- The data that is exchanged by the web service**

Figure 169: Web Services Definition Language (WSDL)

The Web Service Definition Language (WSDL) describes the basic characteristics of a Web service. WSDL is an XML format for describing Web services, similar to the XML schema.

The following elements are defined by WSDL:

- The Web service interface (IDL)
- One or more transport protocols that are used to access the Web service (for example, SOAP)
- The service endpoints or URL

Further Information:

- Team: <http://www.w3.org/2002/ws/desc/>
- Specification: [http://www.w3.org/TR/wsdl12/Web Service Definition Language](http://www.w3.org/TR/wsdl12/Web%20Service%20Definition%20Language)

The following figure describes the content and functions of **Universal Description Discovery and Integration** (UDDI).



Universal Description, Discovery and Integration (UDDI)



- Describes how to advertise and discover a Web service
- Differentiates Web service provider, Web service and Web service type
- Contains metadata that can be used to search for services (names, IDs, categories, types, etc.)
- Specifies the interface for Web service registries



Figure 170: Universal Description Discovery and Integration

Universal Description Discovery and Integration (UDDI) is a place to publish and retrieve Web services. It is an optional Web service feature that describes how to advertise and discover a Web service.

UDDI differentiates the Web service provider, Web service and Web service type.

UDDI contains metadata that can be used to search for services (names, IDs, categories, types, and so on).

For more information about UDDI, see <http://uddi.sap.com>.

To create a Web Service Source Systems follow these steps:

1. In the source system tree in the Data Warehousing Workbench, choose *Create* in the context menu for *Web Service*.
2. In the *Logical System Name* field, enter a technical name for the source system.
3. Enter a description for the source system.
4. In the *Type and Release* field, enter the type of source from a semantic perspective



Note: If SAP ships BI Content for a non-SAP source system, a source type and source release are assigned to this content. If you are using the corresponding system, the correct BI Content can only be found if you specify the source type and source release here.

In order to transfer data into BI using a Web service, the metadata first has to be available in BI in the form of a DataSource.

1. You are in the DataSource tree in the Data Warehousing Workbench.
Select the application components in which the DataSource is to be created and choose *Create DataSource*.
2. In the next screen, enter a technical name for the DataSource, select the type of the DataSource and choose *Copy*. The DataSource maintenance screen appears.
3. Go to the *General* tab page.
 - a) Enter descriptions for the DataSource (short, medium, long).
 - b) If necessary, specify whether the DataSource may potentially deliver duplicate data records within a request.
4. Go to the *Extraction* tab page and define the delta method for the DataSource.
 **Note:** DataSources for Web services support real-time data acquisition. Direct access to data is not supported.
5. Go to the *Fields* tab page.
 - Here you determine the structure of the DataSource either by defining the fields and field properties directly, or by selecting an InfoObject as a Template InfoObject and transferring its technical properties for the field in the DataSource. You can modify the properties that you have transferred from the InfoObject further to suit your requirements by changing the entries in the field list.
 - Entering InfoObjects here does not equate to assigning them to DataSource fields. Assignments are 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.
6. Save and activate the DataSource.
7. Go to the *Extraction* tab page.
The system has generated a function module and a Web service with the DataSource. They are displayed on the *Extraction* tab page. The Web service is released for the SOAP runtime
8. Copy the technical name of the Web service and choose *Web Service Administration*.
The administration screen for SOAP runtime appears. You can use the search function to find the Web service. The Web service is displayed in the tree of the *SOAP Application for RFC-Compliant FMs*. Select the Web service and choose *Web Service → WSDL (Web Service Description Language)* to display the WSDL description.

The DataSource is created and is visible in the Data Warehousing Workbench in the application component in the DataSource overview for the Web service source system. When you activate the DataSource, the system generates a PSA table and a transfer program.

Before you can use a Web service to transfer data into BI for the DataSource, create a corresponding InfoPackage (push package). If an InfoPackage is already available for the DataSource, you can test the Web service push in Web service administration.

This is an example of the generated XML code from a Web service.

Exercise 12: Web Service DataSource

Exercise Objectives

After completing this exercise, you will be able to:

- Create your own Web service with the appropriate components
- Check the Web service for completeness and functions

Business Example

You want to use a function module to fill the Delta Queue. The function module should be called by different applications. It should also perform basic data integrity checks.

For this reason, you enhance the function module that was generated for the DataSource with “SOAP connection” with the necessary checks and generate a Web service based on this function module.

You then want to test whether the definition of the Web service is complete and whether it fulfills the functional requirements. To do this, you use the existing tools in the BI system.

Task 1:

Create a DataSource for Web Services **BW350_WS_GR##** in the Data Warehousing Workbench.

1. Create a DataSource for the WebService source system**ZT_WSERVIC**.

<i>DataSource</i>	GR##_BW350_WS
<i>Source System</i>	ZT_WSERVIC
<i>Data Type DataSource</i>	Transaction Data

Use the following descriptions for your DataSource:

<i>Short description</i>	WebService ##
<i>Medium description</i>	WebService DataSource GR##
<i>Long description</i>	WebService DataSource GR##

Field properties should be copied from Template InfoObjects. Use the following Template InfoObjects to determine the fields of your Web Service DataSource::

Continued on next page

OCUSTOMER
OMATERIAL
OCALDAY
ODOC_NUMBER
OQUANTITY
OUNIT
ONET_PRICE
ODOC_CURRCY

Change all field formats to *Internal* and activate your DataSource. Ignore the displayed warnings.

Have a look at the generated function module (see *Extraction* tab) and check the content of the PSA table of your DataSource.

Task 2:

Examine the Web Service which has been generated by your DataSource.

1. Examine the Web Service which has been generated by your DataSource **GR##_BW350_WS**.

Task 3:

Create an InfoPackage for your WebService DataSource and then check your WebService.

1. Create an InfoPackge with description **Get XML Data GR##**for your WebService DataSource **GR##_BW350_WS**.
2. Test the WebService generated by your DataSource **GR##_BW350_WS** using the test functionality of the Web Service Homepage. The Web Service Homepage can be started from the *Web Service Administration* WSADMIN.

Use the following test data for testing your Web Service:

Field	Value
Customer	1000
Material	1400-100
Calday	Today's date in format YYYY-mm-dd
DocNumber	3745

Continued on next page

<i>Quantity</i>	100
<i>Unit</i>	PC
<i>NetPrice</i>	200
<i>DocCurrency</i>	USD

3. Examine the PSA table associated with your Web Service DataSource.

The PSA table should now contain one data record with the sample data used in step 2.

Solution 12: Web Service DataSource

Task 1:

Create a DataSource for Web Services **BW350_WS_GR##** in the Data Warehousing Workbench.

1. Create a DataSource for the WebServicee source system**ZT_WSERVIC**.

<i>DataSource</i>	GR##_BW350_WS
<i>Source System</i>	ZT_WSERVIC
<i>Data Type DataSource</i>	Transaction Data

Use the following descriptions for your DataSource:

<i>Short description</i>	WebService ##
<i>Medium description</i>	WebService DataSource GR##
<i>Long description</i>	WebService DataSource GR##

Field properties should be copied from Template InfoObjects. Use the following Template InfoObjects to determine the fields of your Web Service DataSource::

OCUSTOMER
OMATERIAL
OCALDAY
ODOC_NUMBER
OQUANTITY
OUNIT
ONET_PRICE
ODOC_CURRCY

Change all field formats to *Internal* and activate your DataSource. Ignore the displayed warnings.

Have a look at the generated function module (see *Extraction tab*) and check the content of the PSA table of your DataSource.

Continued on next page

- a) Use the menu *Data Warehousing Workbench* → *Modeling* → *Source Systems* and open the folder *WebService*.
- b) From the context menu of **ZT_WSERVIC** choose the option *Display DataSource Tree* and find your application component using the menu *BW Training* → *BW350 BI Data Acquisition* → *Group##*.
- c) From the context menu of your Application Component *Group##ZT_BW350_GR##* choose the option *Create DataSource* and in the resulting dialog box enter the following values:

<i>DataSource</i>	GR##_BW350_WS
<i>Source System</i>	ZT_WSERVIC
<i>Data Type DataSource</i>	Transaction Data

Click the *Transfer (Enter)*  icon.

- d) On the *General Info* tab, enter a short, medium and long description as given above.the :
- e) On the *Fields* tab, you define the structure of your Web Service DataSource.
 - Enter the first *Template InfoObject* entry and click the *Enter* key on the workstation keyboard.
 - You will be presented with a dialog box asking whether you want to copy the field properties from the InfoObject. Mark the checkbox *do Not Show This Question Again In This Session* and choose the *Copy* button.
 - To insert a new row in the *Field Attributes* area, click the *Insert Row*  icon.

Follow the bulleted steps above for each entry in each row and enter the remaining Template InfoObjects as given above.

- f) Some field lengths and formats have to be adjusted. change all formats to *Internal* by choosing the dropdown in the *Format* field and changing any InfoObject with a *Format* of **External** to **Internal** and then activate the DataSource by choosing the *Activate*  icon.

Proceed by clicking the *Continue (Enter)*  icon.

- g) On the *Extraction* tab, you will find all the generated components of the DataSource object. Look at the Function Module by double clicking on it. This will take you to the ABAP coding of the Function Module which will load data into the PSA table.

Double click on the **IMPORTING** interface structure.

Continued on next page

Click the *Back*  icon twice to return to the DataSource.

- h) Check the PSA table by clicking the *Manage PSA* icon and you will see that the PSA table is blank.

Task 2:

Examine the Web Service which has been generated by your DataSource.

1. Examine the Web Service which has been generated by your DataSource **GR##_BW350_WS**.
 - a) If you have closed your DataSource **GR##_BW350_WS** then reopen it. Then click on the *Extraction* tab.
 - b) On the *Extraction* tab choose the *Web Service Administration* button. This will take you to transaction WSADMIN.
 - c) Expand the folder **(/BIC/CQGR##... or similar)** containing your WebService and select your WebService.

Then choose the **WSDL** view of your WebService by clicking the *WSDL* icon.

- d) In the dialog “Settings for WSDL Generation” select the radio button *Document Style* and choose the *Continue (Enter)*  icon.

The **WSDL** will be displayed in a browser window. If necessary, login to the BI system using your BI user-id and password.

Once you have seen the generated WSDL close the browser window. You will then be back in the Web Services Administration. Click *Back*  twice to return to the DataSources tree of **ZT_WSERVIC**.

Continued on next page

Task 3:

Create an InfoPackage for your WebService DataSource and then check your WebService.

1. Create an InfoPackge with description **Get XML Data GR##**for your WebService DataSource **GR##_BW350_WS**.
 - a) In the DataSources tree of the WebService source system **ZT_WSREVIC**, find your DataSource **GR##_BW350_WS** in your application component **ZT_BW350-##**.
 - b) From the context menu of your DataSource, choose *Create InfoPackage*.
Enter the *InfoPackage Description*, **Get XML Data GR##**, select your DataSource and click the **Save** icon.
 - c) In your InfoPackage, go to the *Extraction* tab and you will see the Function Module and Web Service generated by your DataSource.
 - d) Go to the *Processing*tab and check the *Automatic Closure of the Request* settings. Accept the default values and save your InfoPackage using the **Save** icon.
 - e) Click the **Back** icon to return to the DataSources display.
2. Test the WebService generated by your DataSource **GR##_BW350_WS** using the test functionality of the Web Service Homepage. The Web Service Homepage can be started from the *Web Service Administration* WSADMIN.

Use the following test data for testing your Web Service:

Field	Value
<i>Customer</i>	1000
<i>Material</i>	1400-100
<i>Calday</i>	Today's date in format yyyy-mm-dd
<i>DocNumber</i>	3745
<i>Quantity</i>	100
<i>Unit</i>	PC
<i>NetPrice</i>	200
<i>DocCurrcy</i>	USD

- a) In the DataSource display locate your WebServices DataSource **GR##_BW350_WS** and from the context menu choose *Change*.

Continued on next page

On the *Extraction* tab of your InfoPackage, click the *Web Service Administration* button.

This will take you to transaction WSADMIN.

- b) Locate and highlight your WebService and then click the *Web Service Homepage* icon.
- c) In the dialog *Settings for WSDL Generation* select the radio button *Document Style* and choose the *Continue (Enter)*  icon.

The Web Service Homepage will be displayed in a browser window. If necessary, login to the BI system using your BI user-id and password and then clicking the *Submit* button.

- d) On the Web Service Homepage, click the option *Test* in the Navigation bar.

In the next view, click the structure of your Web Service and you will be presented with the visualization of your XML interface which is input ready.

- e) Enter the following values in the appropriate fields:

<i>Customer</i>	1000
<i>Material</i>	1400-100
<i>Calday</i>	Today's date in format YYYY-mm-dd
<i>DocNumber</i>	3745
<i>Quantity</i>	100
<i>Unit</i>	PC
<i>NetPrice</i>	200
<i>DocCurrcy</i>	USD

Click the *Send* button.

- f) Close out of the display of your Web Services Homepage to return to BI. You will be in the Web Services Administration area.

Click the *Back*  icon to return to the DataSource display.

3. Examine the PSA table associated with your Web Service DataSource.

Continued on next page

The PSA table should now contain one data record with the sample data used in step 2.

- a) In the display of your Web Services DataSource, choose the *Manage PSA* button. There should be one visible (yellow) request.
- b) Mark the visible request and choose the *PSA Maintenance* button and then click the Enter  icon to proceed.

You should see the entered test data in your PSA table.



Lesson Summary

You should now be able to:

- Describe the important business benefits provided by Web services
- Discuss the significance of Web service standards
- Describe how to implement BI Web services for XML data load

Lesson: XML Data Acquisition Using SAP PI

Lesson Overview

This lesson describes how SAP Exchange Infrastructure (SAP PI) can be integrated with BI for acquiring XML data.



Lesson Objectives

After completing this lesson, you will be able to:

- Gain an overview of the structure and terminology of an SAP PI implementation
- Give examples for SAP PI and BI integration scenarios
- Describe the technologies with which BI and SAP PI can work together
- Describe the implementation of an SAP PI / BI connection

Business Example

You have a heterogeneous system landscape with many different types of source systems (SAP and non-SAP systems). Data exchange between the systems is multidirectional.

You want install a central system for the central distribution of data. This system should standardize and clean the data, and then distribute it to the target systems such as BI.

SAP Solution to Integration Technology

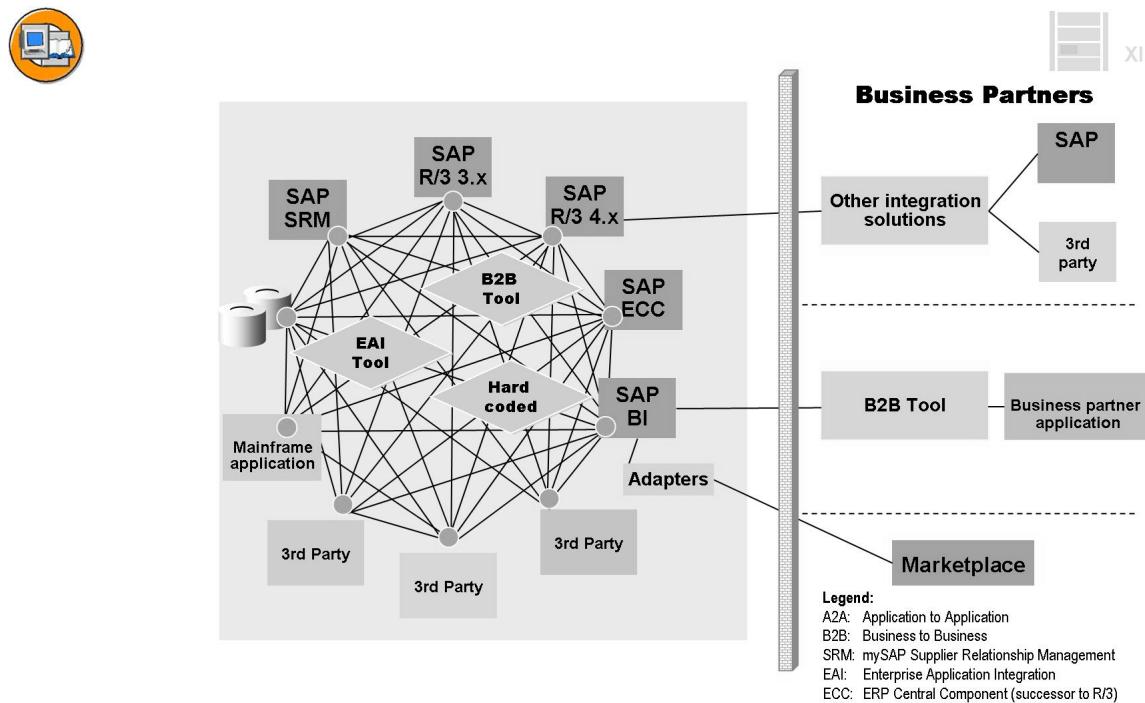


Figure 171: Today's Grown Integration Landscapes

The following statements describe some of the problems associated with the current state of integration landscapes:

- Individual, point-to-point integration, using all available technologies
- Different approach to integration for each purpose (A2A, B2B, BPM, industry standards, and so on)
- Patchwork of integration solutions
- No centralized knowledge or management of integration available
- Grown infrastructure, not adaptable, expensive to maintain
- Considerable costs associated with the integration and upgrade of application components

With the SAP Exchange Infrastructure and collaborative solutions, SAP approaches the integration challenge from a different angle. The concept is based on the availability of a runtime infrastructure with which heterogeneous systems can be combined at the same time with fewer connections. This enables you to link applications so that messages can be transferred from one application to the other, and a centralized store of integration knowledge exists.

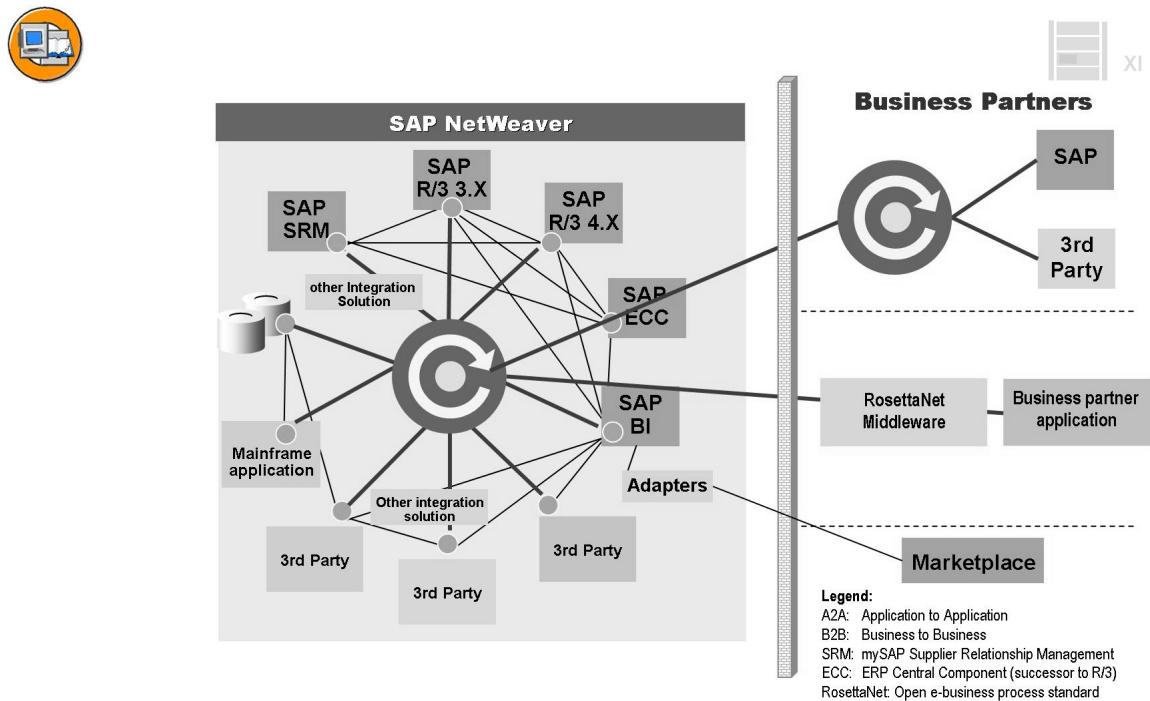


Figure 172: Integration with SAP Exchange Infrastructure (SAP PI)

SAP Exchange Infrastructure (SAP PI) is open and flexible. It uses Web standards such as Web Services Description Language (WSDL), XML Schema Definition Language (XSD), and SOAP messaging for describing objects and communicating with other systems.

SOAP is the SAP technology for connecting SAP Exchange Infrastructure (SAP PI) and SAP NetWeaver Business Intelligence (BI).

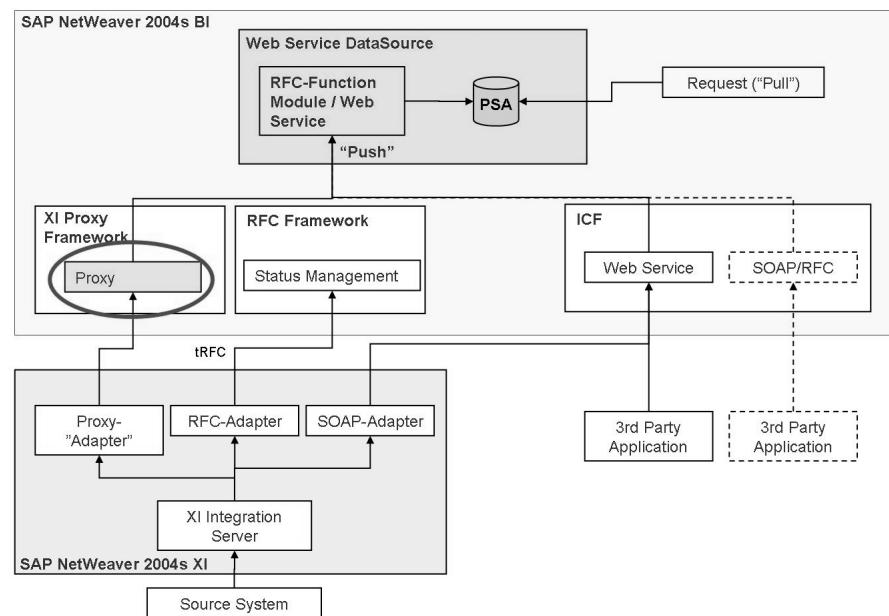


Figure 173: XML Data Acquisition Architecture using SAP PI (Proxy Framework)

The SAP Exchange Infrastructure can be connected to various SAP and non-SAP systems. In the following example of an integration landscape at a customer site, an SAP R/3 3.x (for example R/3 3.1I), an SAP R/3 4.x (for example R/3 4.0B), a third-party system and a new mySAP solution (for example, mySAP SRM) are integrated using the SAP Exchange Infrastructure. The Integration Server also handles the connection to different business partners and to a public marketplace.



Business Content Scenario beyond mySAP SRM 3.0 Global Spend Analysis

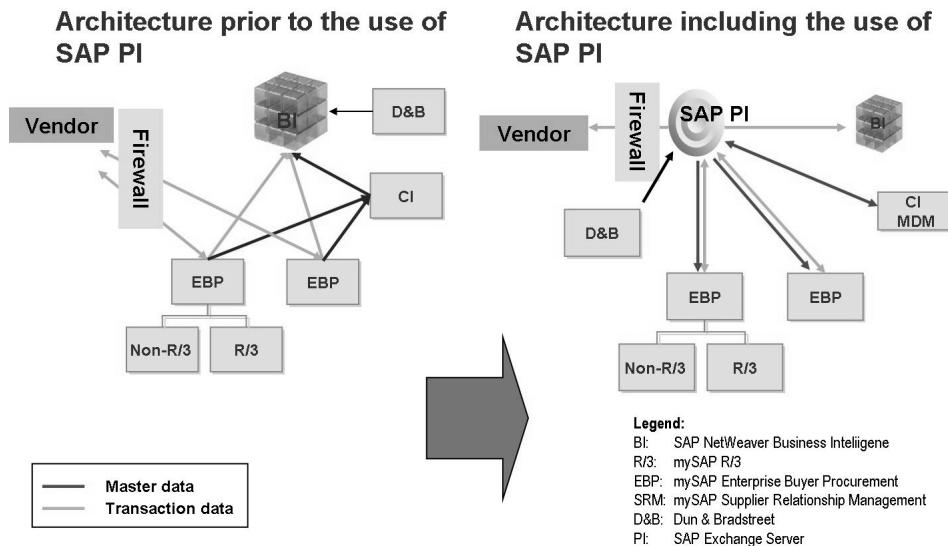


Figure 174: SAP PI Used for Connecting Multiple Systems

Without SAP PI, the integration architecture has the following characteristics:

- Point-to-point connection is provided from the vendor to the EBP systems across company boundaries.
- All the transaction and master data is loaded directly to BI.
- Vendor data and product categories are consolidated in BI and enriched with information from Dun & Bradstreet.
- Vendor data and product categories are consolidated by the Content Integrator (CI).
- Key mapping data is loaded from the Content Integrator (CI) to BI.

With SAP PI, the integration architecture has the following characteristics:

- SAP PI is used to exchange data (purchase orders, purchase order changes, and invoices) between the vendor and the SAP SRM systems across company boundaries.
- The exchanged data is copied to SAP PI and pushed to BI; there is no point-to-point connection to the EBP.
- SAP ERP transaction and master data can be loaded directly to BI.
- Vendor data and product categories are consolidated in SAP MDM and can be enriched with Dun & Bradstreet information in BI.

Using the SAP PI/BI connection offers the following benefits:

- For analysis purposes, the data is also distributed to BI through SAP PI. Data extraction from the mySAP SRM source systems to BI is therefore not required.
- Data consistency and the data details are fully maintained as the data is distributed from system to system.
- The advantages of SAP PI are used to connect mySAP SRM to vendors.



Business Content Scenario – Retail Store Connectivity Scenario

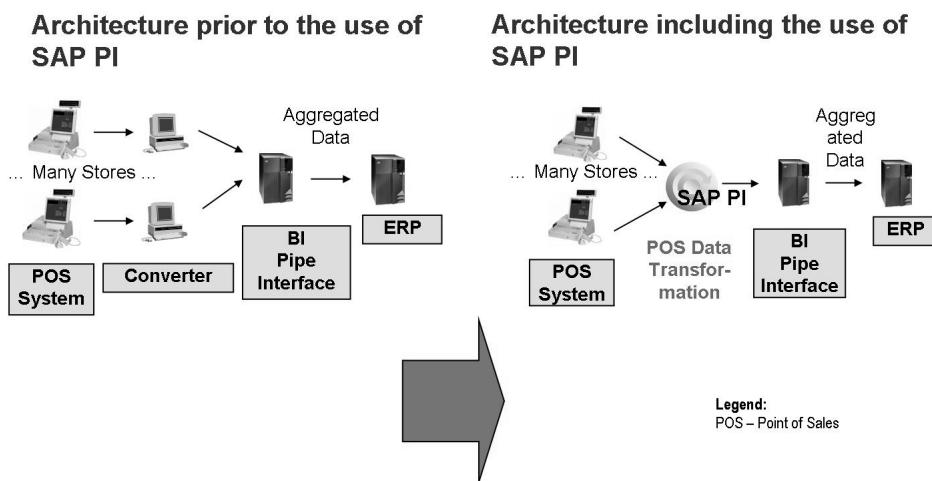


Figure 175: SAP PI and BI Integration Scenarios

Architecture prior to the use of SAP PI

- The POS data has to be converted from the format of the source system to the format of the target system.
- Many sources (stores) are connected to BI.
- The daily store data is placed on a file server from a specific time onwards.
- The BI system extracts the data from the file server using a file upload (pull mechanism).

Architecture including the use of SAP PI

- SAP PI constitutes a central point for calling POS information as mass data from (third-party provider) store systems using an ARTS/IX compliant, open, industry-specific interface.
- Using SAP PI as only one additional source (system) for BI (as opposed to many source systems).
- Calling the XML data from the store file server automatically.
- The store is directly responsible for the availability of the data

Reason for using the SAP PI/BI connection in this scenario

- The data must be pushed to BI
- SAP PI supports the full quality of service „Exactly once in order (EOIO) in push scenarios (necessary to ensure transactional integrity in BI)
- Stores deliver the data according to ARTS/IX and SAP PI supports ARTS/IX



Hint: If the stores deliver the data as flat files they can be transformed to XML format simply using SAP PI.

- Compliant with the SAP NetWeaver architecture specification, fully enabled for future growth of SAP offerings and the NetWeaver platform

Other characteristics of integration scenarios are listed in the following figure and text.



- **Source system landscape**
 - How many source systems will be connected to BI?
 - Do the source systems communicate with each other, or is a just one connection to BI required?
 - Are the source systems of the same type?
 - What is the cost of connecting one (additional) source system to BI for each integration scenario?

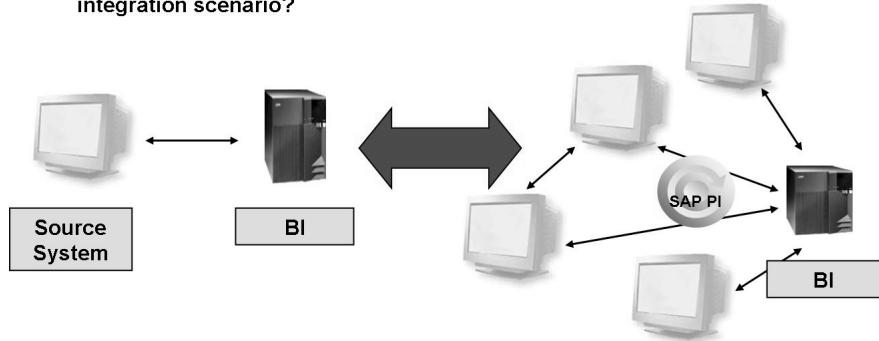


Figure 176: Characteristics of Integration Scenarios

- **Communication**
 - Is the source system able to send data at all?
If the answer is YES: Which technology is to be used and which is supported by BI or SAP PI technology?
If the answer is NO: Think of options for transferring data to BI, for example, calling data through SAP PI or by flat file upload to BI
- **Persistence**
 - Is requested data stored persistently on the database or on the file system for the source system?
If the answer is YES: Database or file adapter in SAP PI, standard methods in BI
If the answer is NO: Standard scenario for SAP PI, push using XML to BI
- **Implementation cost**
 - Do integration scenarios between BI and source systems or SAP PI and source systems exist before the start of the project?
- **ETL functions**
 - If a great deal of effort is required to transform the data from the source system: Which ETL functions are required to transfer data to BI (for example, when transforming and mapping the data)? What is the cost of implementing transformations in a particular integration scenario?

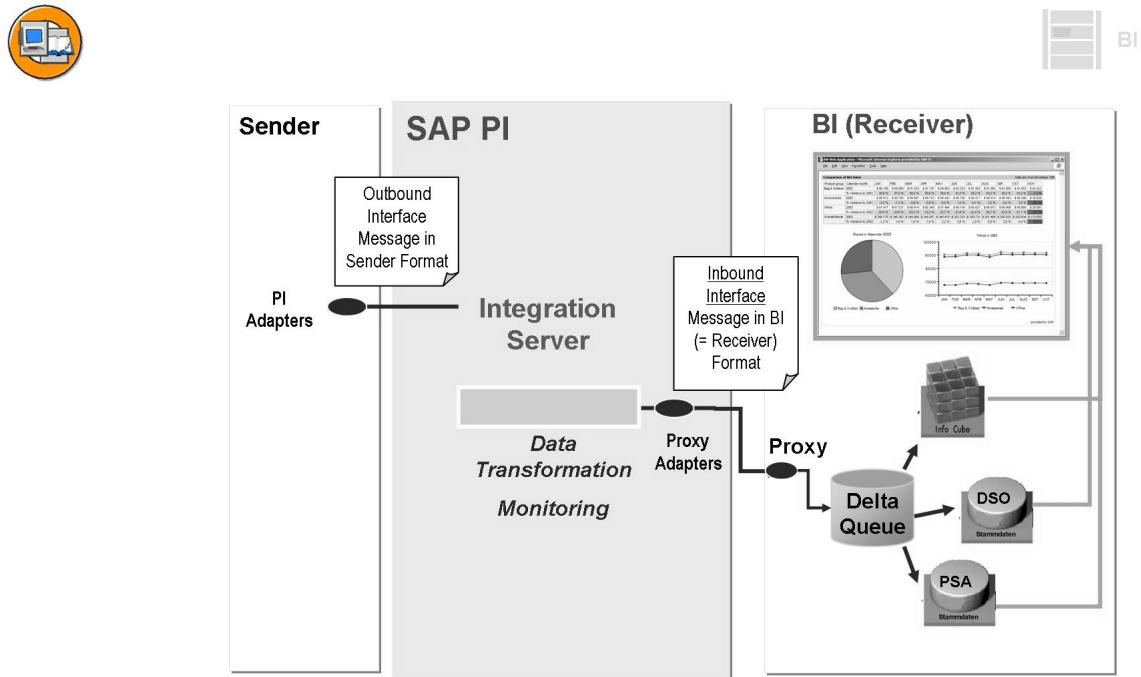


Figure 177: Connecting SAP PI and BI - Overview 1

A major factor for the type of connection is the quality of service provided by the different technologies

- BE (Best Effort): The message is sent synchronously. This means that the sender system waits for a response before it continues processing. Messages are not retained by the Integration Engine in synchronous message processing. Once a message has been processed in the target system, an implicit database commit (SOAP Communication) follows.
- EO (Exactly Once): The message is sent asynchronously. In this case; this means that the sender system does not wait for a response before continuing processing. The Integration Engine guarantees that the message is sent and processed only once (RFC Adapter).
- EOIO (Exactly Once In Order): In addition to the Exactly Once method, messages with the same queue names (supplied by the application) are delivered in the same sequence as they were sent from the sender system. Message processing is asynchronous in this case (proxy communication).

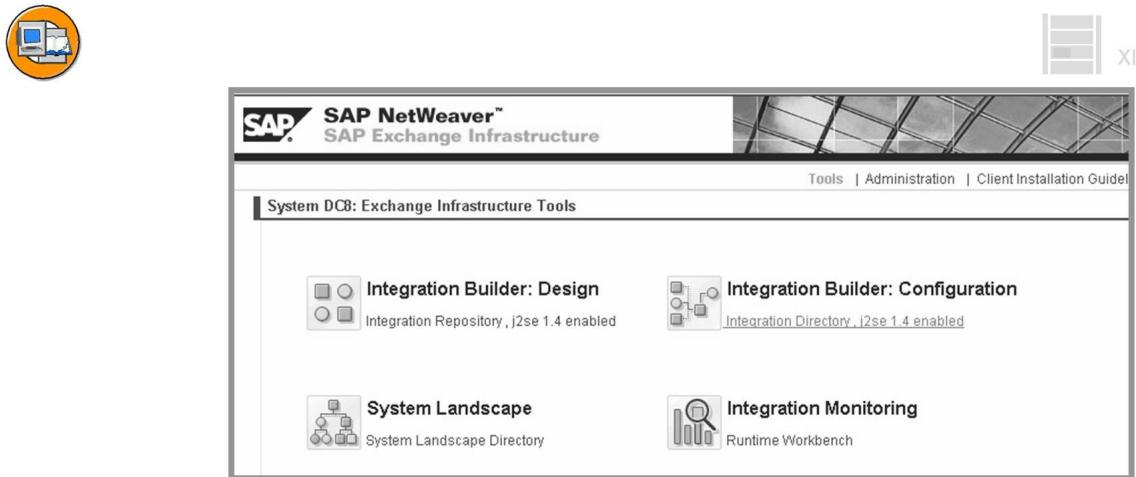


Figure 178: Connecting SAP PI and BI - Overview 2

System Landscape Directory (SLD)

- Enables you to describe products, software components, logical systems, and technical systems. SAP Exchange Infrastructure accesses this information at design time, configuration time, and runtime.

Integration Builder: Design

- During the *design phase*, you document the entire collaborative process and determine which interfaces are required. You can either define new system-independent interfaces to be implemented at a later point in time (*outside-in development*) or work with functions that already exist in the systems (*inside-out development*). In this phase you design the logical collaborative process by describing in a specific role the message exchange between the application components. This description is still not specific to any particular installed system.

Integration Builder: Configuration

- During the *configuration phase*, you configure the collaborative process for a specific system landscape. For example, you define conditions for the message flow and select design objects that meet your requirements.

Integration Server (Monitoring): Runtime

- Central “distribution engine” for messages in the SAP Exchange Infrastructure at runtime. All systems that use SAP Exchange Infrastructure to communicate use this server to exchange messages. Using the configuration data from the Integration Directory, the Integration Server decides to which receivers it must send the message and whether mapping needs to take place beforehand.

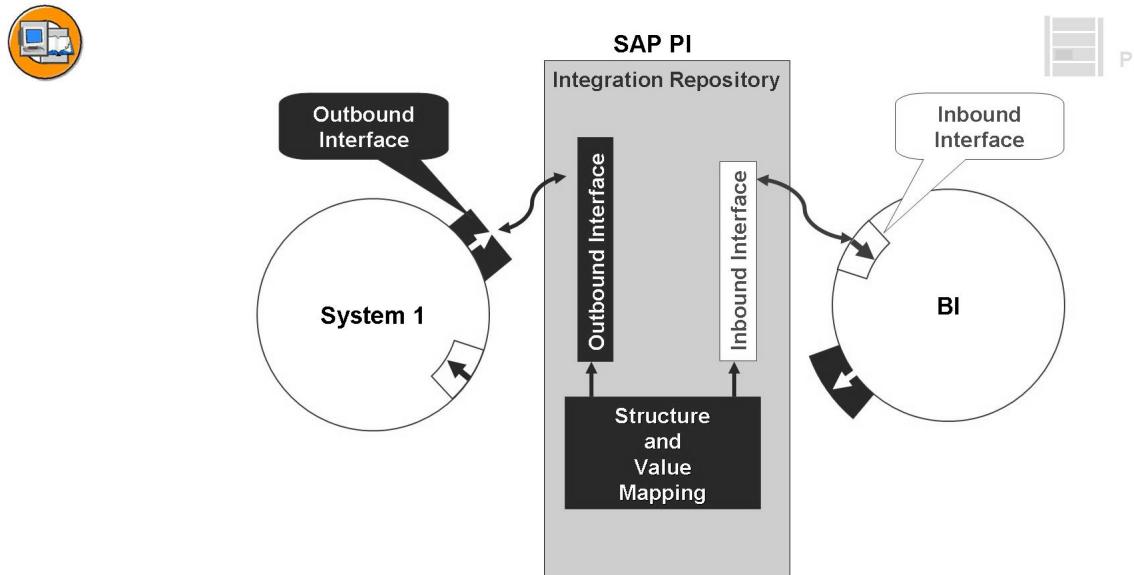


Figure 179: Integration Repository (Design Time)

Definition Interface: Platform-independent description (interface) for exchanging messages.

Communication in SAP PI is based on interfaces. This means that messages are normally created by calling an interface.

The Integration Repository contains the interfaces for the systems you are using. In some instances you can import them to the Integration Repository, in other instances you have to create them manually. Save the mapping rules for structure and value mapping for each external interface. There is a graphical mapping tool for this purpose. Alternatively, you can create mapping programs in Java or XSLT.



● Mapping

- Transformation from one message structure to another
- Transformation rules are defined by a *mapping program*

Figure 180: Mapping Concepts

There are different mapping techniques in PI.

- Message mapping
 - Graphical design and test environment
 - Queue-based model with the option of handling large documents
 - Extensible through user-defined functions in Java
- **XSLT (eXtensible Stylesheet Language)**
 - Open standard
 - Portable
 - Extensible through user-defined functions in Java
 - Considerable memory requirement for very large documents
- **Java**
 - Flexibility of Java programming language
 - Java mapping program is responsible for parsing and rendering XML code
- **ABAP**
 - Leveraging the existing ABAP Basis
 - ABAP mapping program is responsible for parsing and rendering XML code



The goal of the Integration Directory is to configure the sender-receiver relationships which will be used at runtime.

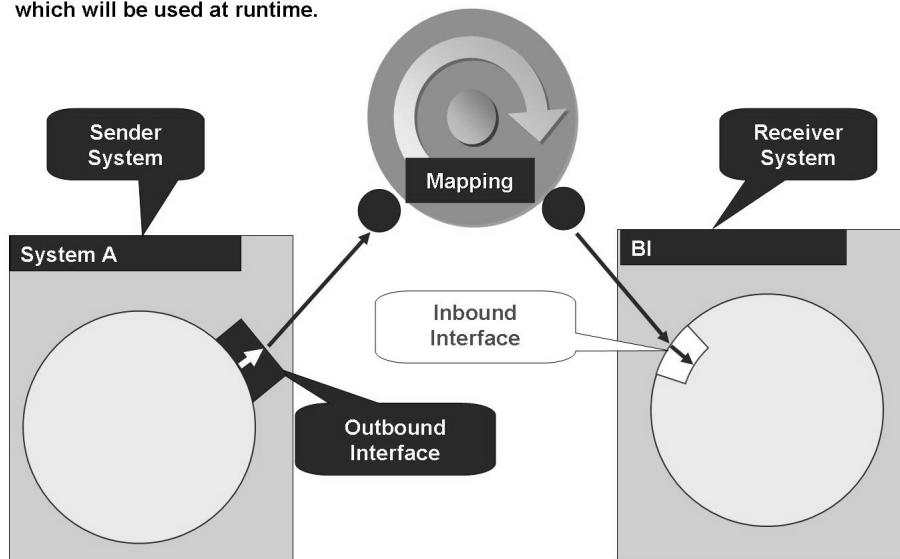


Figure 181: Integration Directory (Configuration Time)

At configuration time, you configure the cross-system processes for an existing system landscape. The configuration describes how the Integration Server is to process inbound messages and to which receivers messages must be sent.

When creating sender-receiver relationships, there are many aspects to consider. Examples:

- Which system should receive the message?
- Which interface should receive the message?
- Which transport mechanism will be used?
- Which security requirements exist?

The PI Integration Repository is used to determine these and other parameters so that messages can be processed properly when they arrive at the Integration Server.

When the IS receives a message, it checks the header to determine the sender. Using this information, the recipient information is taken from the configuration and the message is passed on with the required editing.

Separation of business application logic from the PI connection

Outside-in development approach

- Out-of-the-box integration without adapter
- For applications based on SAP Web AS 6.20 or above (ABAP and Java)

Proxy framework makes the technical details transparent for the application developer

- Transform language-specific data structures into XML and vice-versa
- Ensure technical connectivity with the Integration Engine, guaranteed transfer

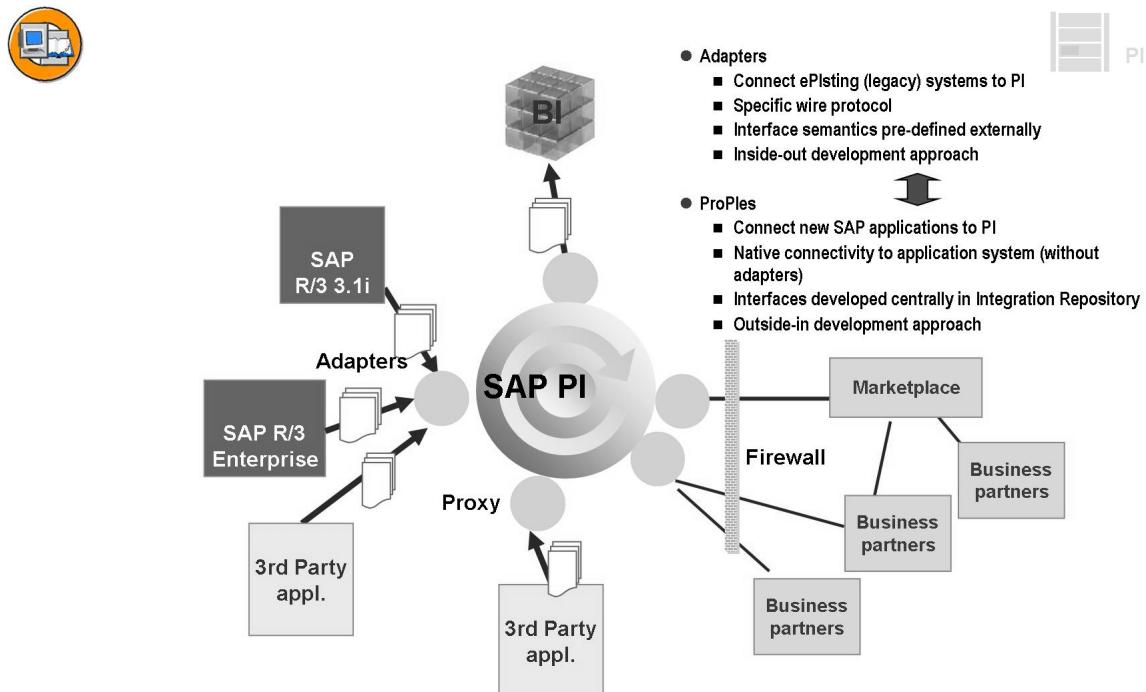


Figure 182: Connecting System with SAP PI - Proxies vs. Adapters

It is important to realize that proxies and adapters are the 2 alternatives for connecting PI to an application system.

The interface semantics cannot normally be changed for an existing system (any external system or even a traditional SAP system that only communicates with RFC and IDoc). In many cases you also have to use a specific, proprietary wire protocol. This is the purpose of the adapters. This is also the precondition for outside-in integration or appended integration (see the next figure).

With new SAP applications (ABAP or Java) the interface development process has changed. The interface is designed centrally in the Integration Repository. A proxy is generated in ABAP or Java from the interface definition. The proxy is deployed on the application system, and the business application is built around it. This is the precondition for inside-out integration (integration by design).

A proxy is a fragment of code in ABAP or Java which serves these purposes:

- Converting the data structures (ABAP or Java) into XML messages and vice versa
- Establishing a connection to the PI Integration Server

A proxy makes such technical details transparent for the application developer.

Note that no specific adapter configuration is required for proxy communication. From a technical point of view however, the proxy runtime itself resides on the adapter framework. The point is that no protocol conversion is necessary for communicating with PI using proxies.

You can find more information on the use of proxies in the PI Fundamentals course or in the online documentation.

Adapters can be arranged according to their function:

- **Application adapters**

SAP applications (RFC, IDoc), Siebel, Oracle, PeopleSoft applications

- **Technical adapters**

File systems (File/FTP) ,RDBMS systems (JDBC), Messaging systems (JMS), Web Services (SOAP), Mail servers (SMTP), SAP Business Connector (SAPBC), Marketplaces

- **Industry standard adapters**

RosettaNet, CIDX, UCCnet

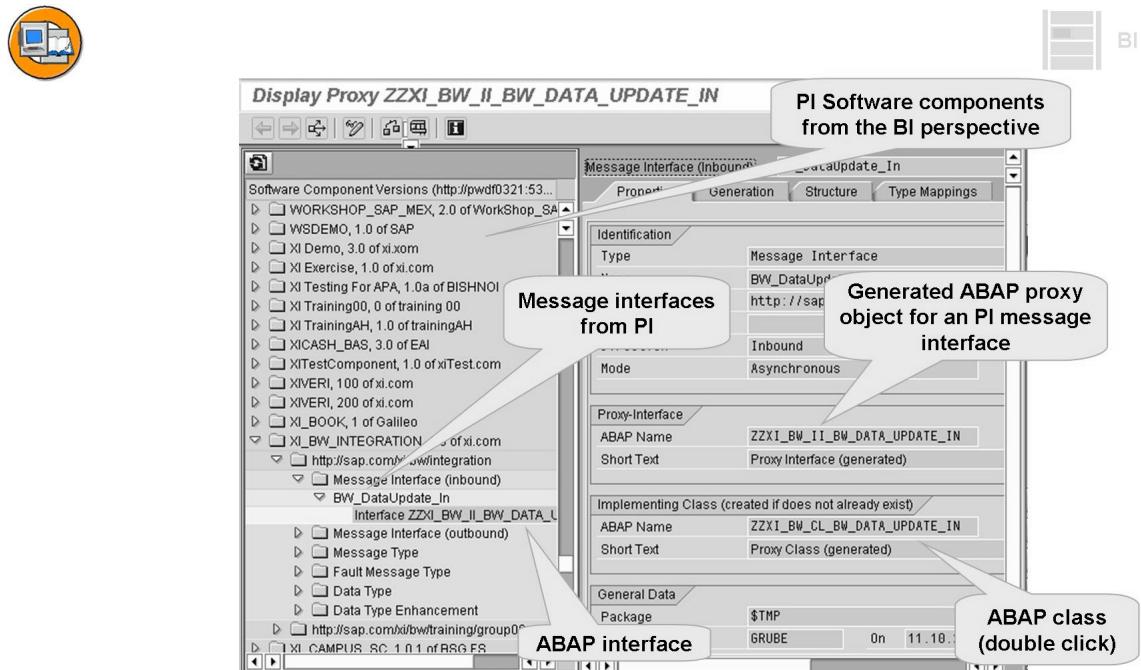
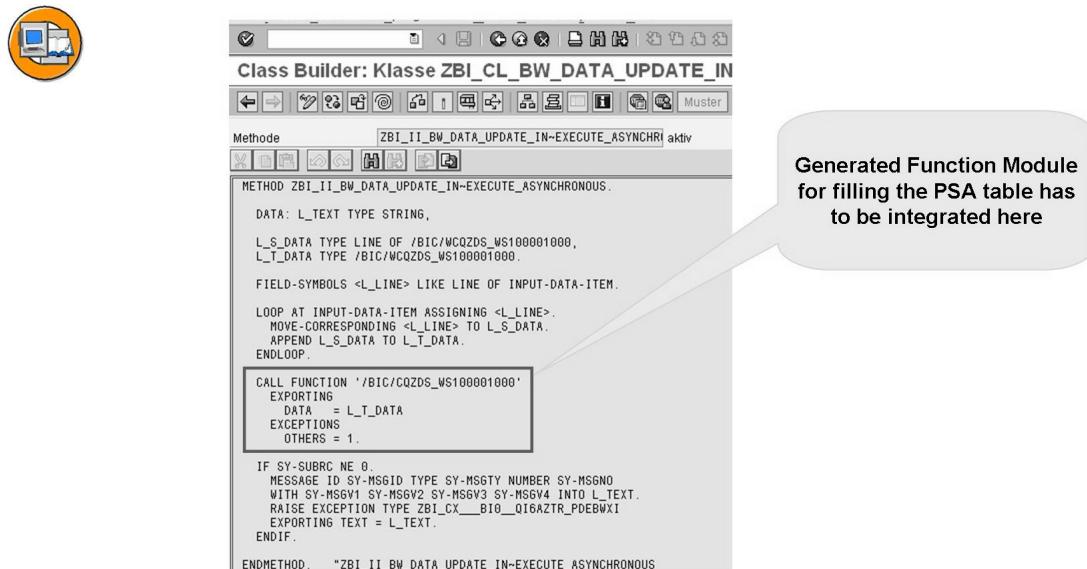


Figure 183: Transaction in BI: SPROXY

In transaction SPROXY, you will find all the objects and tools required to generate the proxy.

To generate the proxy, proceed as follows:

1. Search for your inbound message interface (in the BI system). To the left you will see the tree structure of the interfaces from the PI Integration Repository.
2. If no proxy exists for your message interface you can generate this by way of the context menu.
3. After this you will see details of the generated proxy on the right of the screen. From here you can also navigate to the implementing class for the proxy and implement the coding for the proxy (see the example on the following page).



```

Class Builder: Klasse ZBI_CL_BW_DATA_UPDATE_IN
Methode ZBI_IT_BW_DATA_UPDATE_IN-EXECUTE_ASYNCRI aktiv
METHOD ZBI_IT_BW_DATA_UPDATE_IN-EXECUTE_ASYNCRI.
DATA: L_TEXT TYPE STRING,
L_S_DATA TYPE LINE OF /BIC/WCQZDS_WS100001000,
L_T_DATA TYPE /BIC/WCQZDS_WS100001000.
FIELD-SYMBOLS <L_LINE> LIKE LINE OF INPUT-DATA-ITEM.
LOOP AT INPUT-DATA-ITEM ASSIGNING <L_LINE>.
MOVE-CORRESPONDING <L_LINE> TO L_S_DATA.
APPEND L_S_DATA TO L_T_DATA.
ENDLOOP.
CALL FUNCTION '/BIC/CQZDS_WS100001000'
EXPORTING
  DATA = L_T_DATA
EXCEPTIONS
  OTHERS = 1.
IF SY-SUBRC NE 0.
MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4 INTO L_TEXT.
RAISE EXCEPTION TYPE ZBI_CX__BIO_QI6AZTR_PDEBWXI
EXPORTING TEXT = L_TEXT.
ENDIF.
ENDMETHOD. "ZBI_IT_BW_DATA_UPDATE_IN-EXECUTE_ASYNCRI

```

**Generated Function Module
for filling the PSA table has
to be integrated here**

Figure 184: Implementing the Proxy

Now that we have explored the different ways in which XML data acquisition can be accomplished in BI, the following graphic offers suggestions as to the appropriate approach.



Web AS SOAP Service	Web Service	SAP PI
Based on DataSource with "SOAP Connection"		
Data is written to DeltaQueue of BI system		
Support of Internet standards such as SOAP protocol and XML		
Standard out-of-the-box functionality	Flexibility in coding	Heterogeneous system landscape
Unsuitable for mass data loading processes	Can be easily accessed by every user (application)	SAP PI is already part of the system landscape
No guarantee of single transmission	Transmission depends on the implementation of the function module	Guaranteed single transmission
		Many sources provide data for the BI system
		Mass data transfer option

Figure 185: Which Extraction Mechanisms Should You Use?



Lesson Summary

You should now be able to:

- Gain an overview of the structure and terminology of an SAP PI implementation
- Give examples for SAP PI and BI integration scenarios
- Describe the technologies with which BI and SAP PI can work together
- Describe the implementation of an SAP PI / BI connection



Unit Summary

You should now be able to:

- Explain the underlying standards for XML-based extraction
- Name the necessary objects in the BI system
- Implement the common framework for bringing together the different extraction mechanisms
- Describe the function of the SOAP service DataSource 3.x in the SAP Web Application Server
- Explain how the SOAP service DataSource 3.x enables data acquisition from an application
- Describe the important business benefits provided by Web services
- Discuss the significance of Web service standards
- Describe how to implement BI Web services for XML data load
- Gain an overview of the structure and terminology of an SAP PI implementation
- Give examples for SAP PI and BI integration scenarios
- Describe the technologies with which BI and SAP PI can work together
- Describe the implementation of an SAP PI / BI connection

Unit 9

Data Acquisition with Third-Party ETL Tools

Unit Overview

In this unit, we will describe the tasks and functions of third-party ETL tools, along with the procedure for connecting a third-party ETL tool to BI.



Unit Objectives

After completing this unit, you will be able to:

- Describe the functions and areas in which third-party ETL tools can be used
- Explain what staging BAPIs are
- Describe the procedure for connecting a third-party ETL tool to *BI*

Unit Contents

Lesson: Data Transfer with Third-Party ETL Tools 448

Lesson: Data Transfer with Third-Party ETL Tools

Lesson Overview

This lesson describes the tasks and functions of third-party extraction, transformation and loading (ETL) tools, and how to connect a third-party ETL tool to BI.



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the functions and areas in which third-party ETL tools can be used
- Explain what staging BAPIs are
- Describe the procedure for connecting a third-party ETL tool to BI

Business Example

Your company wants to transfer large datasets from external systems to BI. The datasets are located on various system platforms and have different data formats. Complex transformations are required to bring the data into line with BI structures. It is your job to connect a third-party ETL tool to BI to carry out the data staging process.

Third-Party ETL Tools

Third-party extraction, transformation and loading (ETL) tools can also be used for data extraction. The functions of these ETL tools are specific to the individual tool. You will find information about them in the documentation provided by the respective provider.

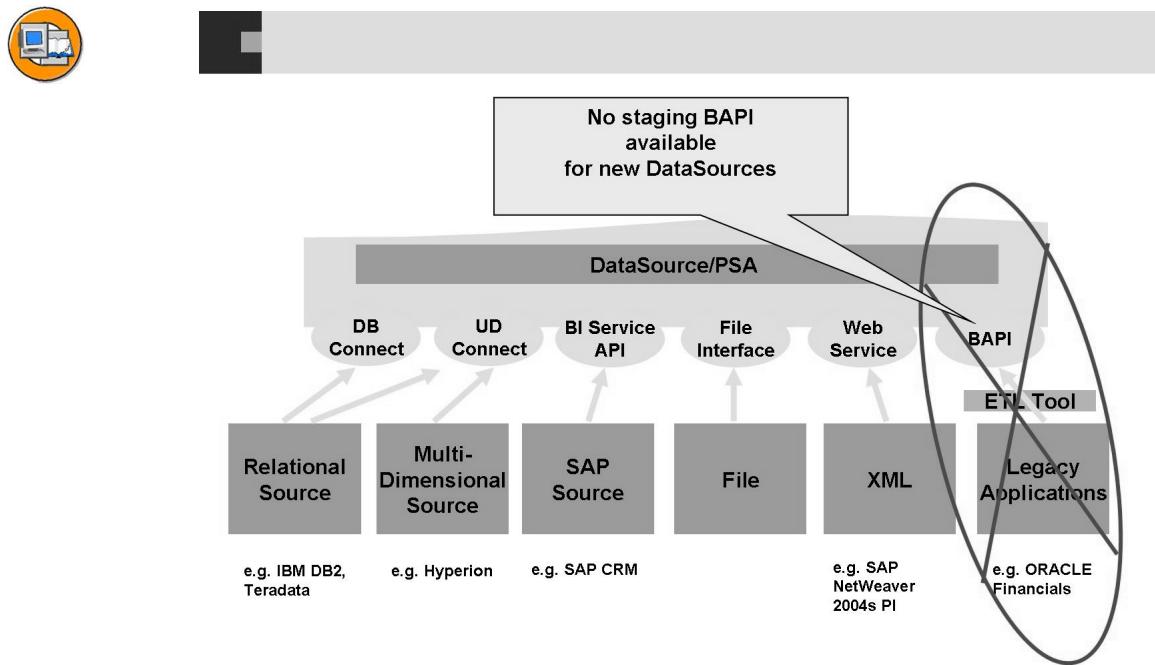


Figure 186: Data Transfer with Third-Party ETL Tools



Hint: Various third-party providers are certified for connection to BI. For an overview of the certified ETL tools, go to the SAP Service Marketplace at www.service.sap.com/bi and choose *Partner → Complementary Software Partner*.

To enable you to extract data and metadata from external systems, BI provides open interfaces, **Staging BAPIs (Business Application Programming Interfaces)**. These interfaces allow you to connect different ETL tools to BI.



Note: The Staging BAPIs only work with **DataSource 3.x** and *not* the new **DataSources**.

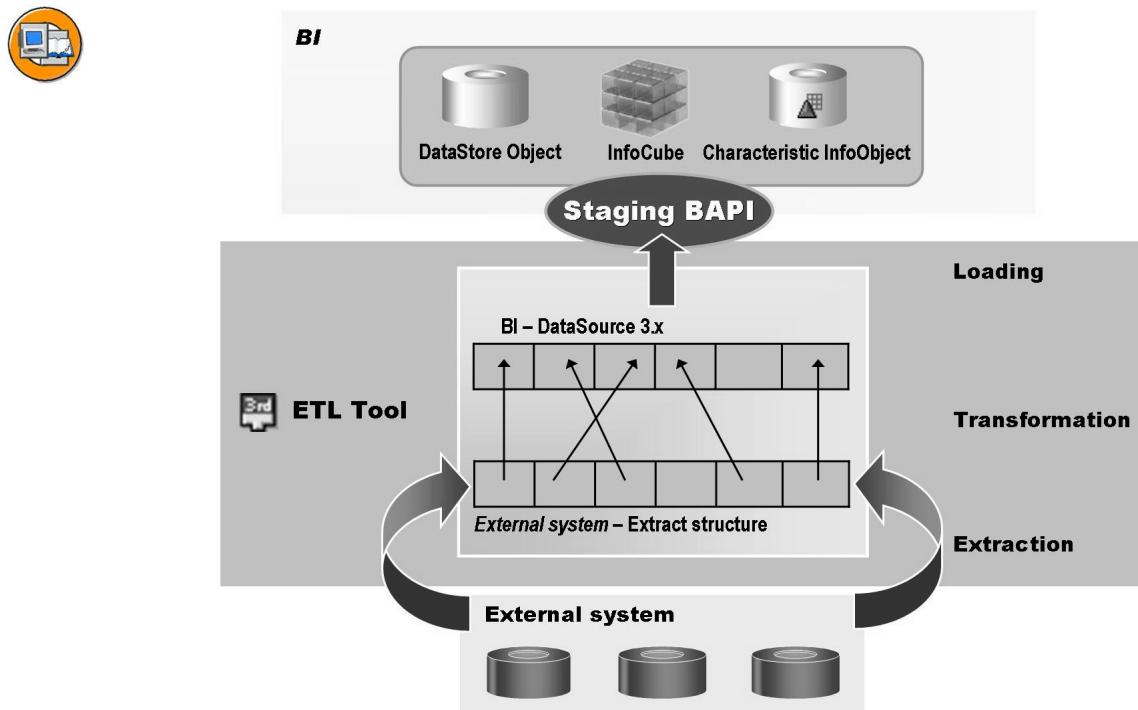


Figure 187: ETL Tools

You can define or update metadata manually in the maintenance screen for the transfer structure in BI. If you access BAPIs with an ETL tool, this tool can automatically read or define metadata from external systems without a specific request from BI, and then transfer it to BI using the BAPIs. BI provides additional interfaces that ETL tools can use to create metadata in the BI system.

Data can be transferred by means of a data request from BI or triggered by the ETL tool with the help of BAPIs. The ETL tool loads the data from the external system and transforms it into the appropriate BI format. Note that the setup of the transfer structure and the setup of the data structure for the ETL tool are similar. Data transformations for the technical cleanup (such as date conversions) should already have been done at ETL tool level.

ETL Tools: Functions

ETL tools enable you to load data and metadata from external systems to the BI system. To fulfill this task, ETL tools normally provide the following functions, although these can vary from provider to provider:



- **Connecting Different Platforms**
- **Supplying Transformation Logic**
- **Design of ETL Processes**
- **Execution of ETL Processes**
- **Administration of ETL Processes**
- **Binding to the BI System**

Figure 188: ETL Tools: Functions

The following text describes the ETL functions in detail:

Connecting different platforms

ETL tools give you the option of linking to standard database systems (native or ODBC (Open Database Connectivity)), different file systems or XML data sources, for example. In addition, some third-party providers offer specialized LoadPackages that enable you to extract data from eBusiness applications (Siebel, i2, and so on).

Staging transformations

ETL tools offer functions for string processing and for carrying out complex calculations, for example. Various data sources can also be integrated by reading data from other sources in an extraction process. You can clean up data according to certain criteria and transfer them in a file containing records with errors. Data can also be aggregated.

Design of ETL processes

You can use ETL tools to model data staging processes (usually supported with graphics too). To do this, you use the transformations described above and the link to different data sources.

Executing ETL processes

Most ETL tools have a transformation engine that executes the ETL processes you have defined. You can schedule jobs and monitor them too.

Administration of ETL processes

You use administration components to manage users and authorizations and maintain metadata (such as tables and structures).

Connecting to the BI system

ETL tool from certified partners can be integrated with the BI system. In addition to the actual loading process, integration also includes the exchange of metadata.

ETL Tools: Area of Usage

The use of an ETL tool brings particular benefits when you want to transfer a large volume of data from external systems to BI and the data comes from heterogeneous sources, requiring complex transformations before it can be

transferred. The only other alternative would be to develop your own tool, which is both time-consuming and cost-intensive. ETL tools have the following advantages over the development of your own tool:

- The graphic modeling and wide-ranging functions make it quicker and easier to develop ETL processes from external systems.
- The extraction processes are uniformly documented, which improves maintenance.
- You can use the ETL tool as a central administration tool for extraction from external systems since data from different external systems is integrated in a standardized environment.

Integration of ETL Tools with BI Using Staging BAPIs

The ETL tool and BI can be integrated using interfaces. Special BAPIs referred to as **staging BAPIs** are used to exchange data and metadata between the ETL tool and BI. The individual BAPIs contain methods for updating and staging metadata and sending extracted data to the BI system. The interfaces are open interfaces and can also be used by customers for their own extraction programs, for example. ETL tools certified by SAP use these BAPIs. You will find more detailed information about these BAPIs in the Interface Repository at <http://ifr.sap.com> or by entering transaction BAPI (BAPI Explorer).

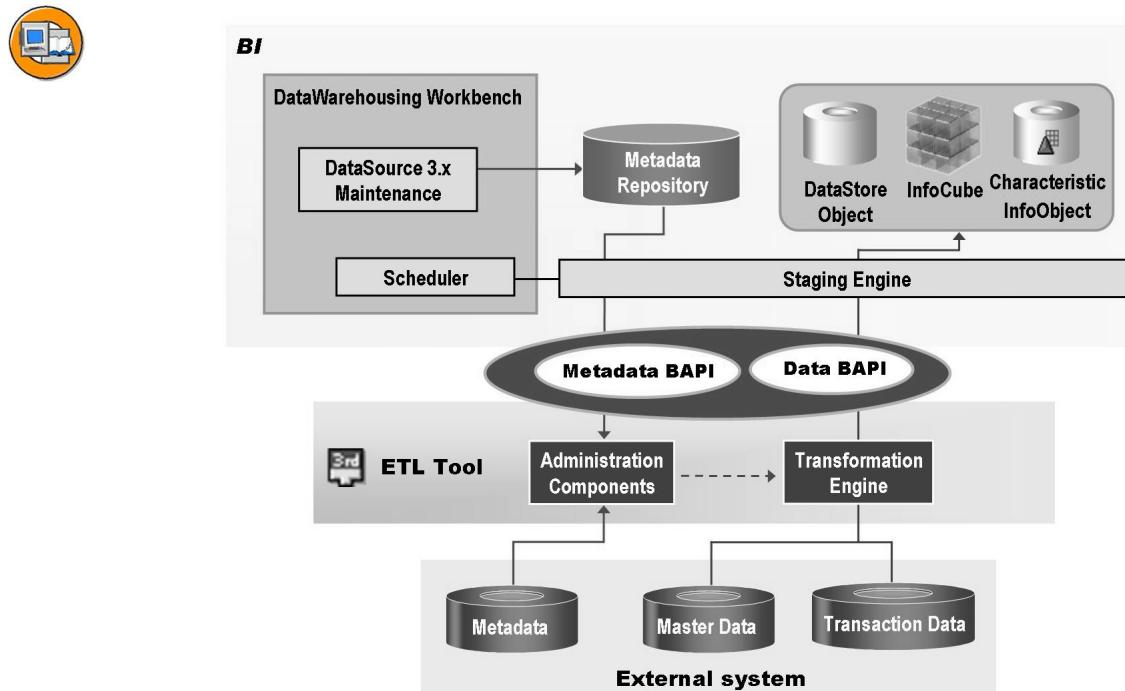


Figure 189: ETL Tools and Staging BAPIs

The BAPI interface is used to communicate the setup of the transfer structure in BI to the ETL tool, for example. The ETL tool, on the other hand, can also create InfoPackages and trigger the loading process directly from outside the system. BAPIs are also used for uploading data from the ETL tool to BI.

ETL Tools: Connection

ETL tools are recognized as the source system by the BI system. You can therefore set up a link to the ETL tool by creating a new source system in the source system tree in the Data Warehousing Workbench.

To link an ETL tool to the BI system, several steps are necessary:

- Choose *Create Source System* in the source system tree.

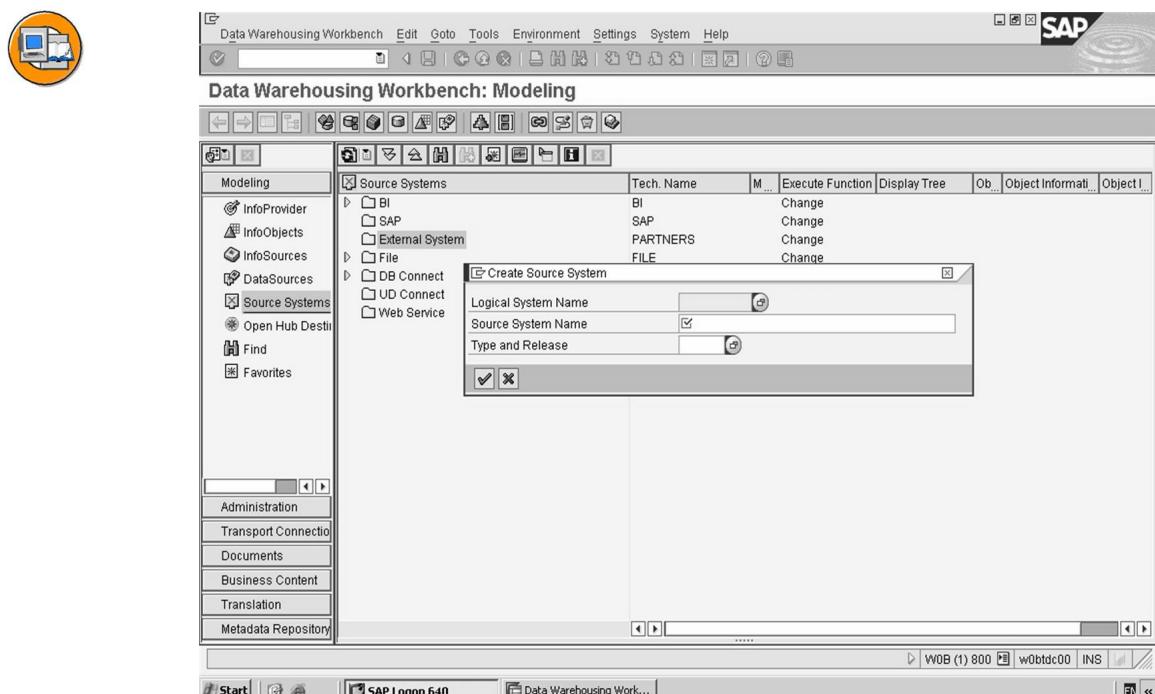


Figure 190: ETL Tool: Create External System

- Choose *External System...* as the source system type.
- Maintain the logical system name.
- In the following screen, choose activation type *Registered Server Program* and enter the program ID.

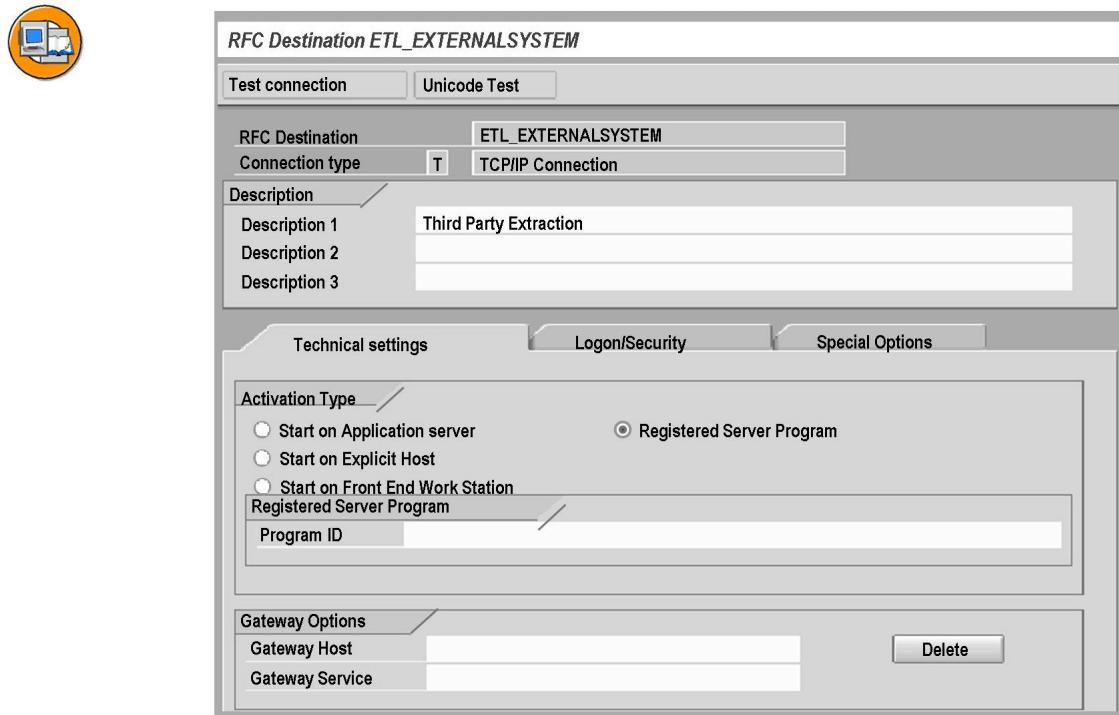


Figure 191: ETL Tool: Connection

- Save your settings and entries for the source system to complete the definition in the BI system.
- When you have defined the connection in the ETL tool too, you can test this using the transaction SM59.
- Define an InfoSource to load the data. InfoSource maintenance and the procedure that follows is the same as when loading data for flat files.



Lesson Summary

You should now be able to:

- Describe the functions and areas in which third-party ETL tools can be used
- Explain what staging BAPIs are
- Describe the procedure for connecting a third-party ETL tool to *BI*



Unit Summary

You should now be able to:

- Describe the functions and areas in which third-party ETL tools can be used
- Explain what staging BAPIs are
- Describe the procedure for connecting a third-party ETL tool to *BI*

Unit 10

Data Mart Interface

Unit Overview

In this unit, data exchange between multiple BI systems and within an BI system is described and their various uses will be worked out.



Unit Objectives

After completing this unit, you will be able to:

- Extract the data from a source BI system to another target BI system

Unit Contents

Lesson: Data Mart Interface	458
Exercise 13: Data Mart Interface	465

Lesson: Data Mart Interface

Lesson Overview

This lesson describes data exchange between multiple BI systems and highlights the various uses for this data exchange.



Lesson Objectives

After completing this lesson, you will be able to:

- Extract the data from a source BI system to another target BI system

Business Example

You want to use the Data Mart interface because the BI system in your headquarters needs aggregated data from the BI systems in the subsidiaries. You also want to evaluate the sales data from an external system using an InfoCube in BI. The external data is staged using flat files in CSV format.

Data Mart Interface

Whereas the data warehouse is an information system that covers an entire enterprise, data marts are area-specific solutions that can be implemented using different system architectures in a data warehouse landscape. Data marts can be used as a development stage in a long-term approach across the whole of the company, allowing rapid implementation with a manageable volume of data. This enables you to realize a company-based data warehouse solution step by step.

As a Service API component, the Data Mart Interface enables data to be updated from one InfoProvider to another.

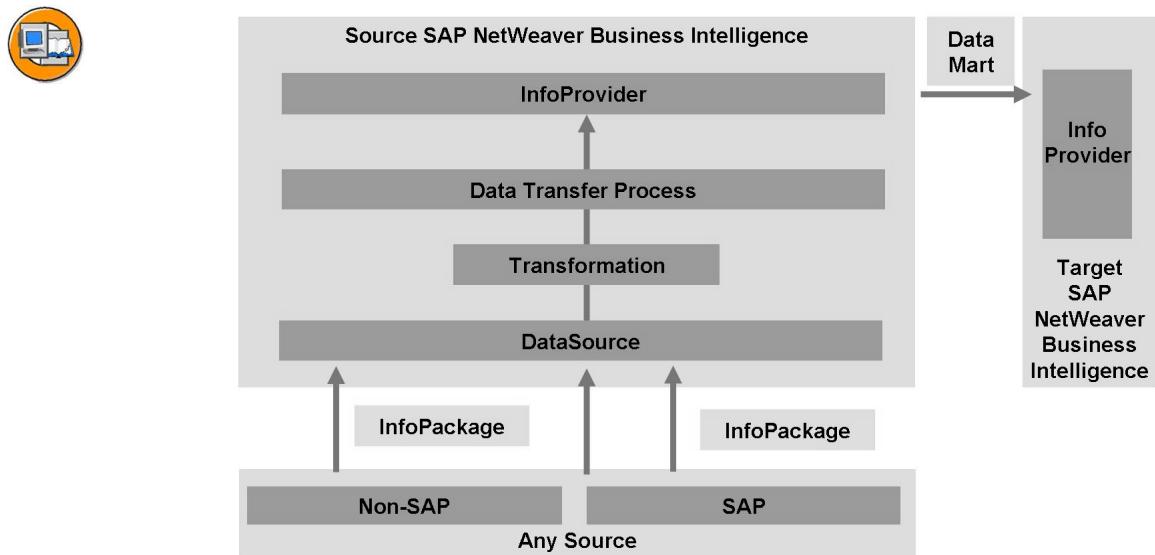


Figure 192: Data Mart Interface

Data exchange between several BI systems is the point of the Data Mart Interface. The system that supplies the data is described as the source BI system; the system that receives the data is the target BI system.

Data Mart Architectures for BI Systems

The Data Mart Interface function shown above can be used to realize the following Data Mart architectures for BI systems.



- **Replication Architecture**
- **Aggregation Architecture**
- **Mixed Architecture**

Figure 193: Data Mart Architecture for BI Systems

Replicating Data Mart architecture

With a **replicating** architecture, also known as Inside-Out architecture, a BI system provides the source data for updating to other BI systems. The data selected and the specific updates to the target BI systems depends on the requirements of the company concerned.

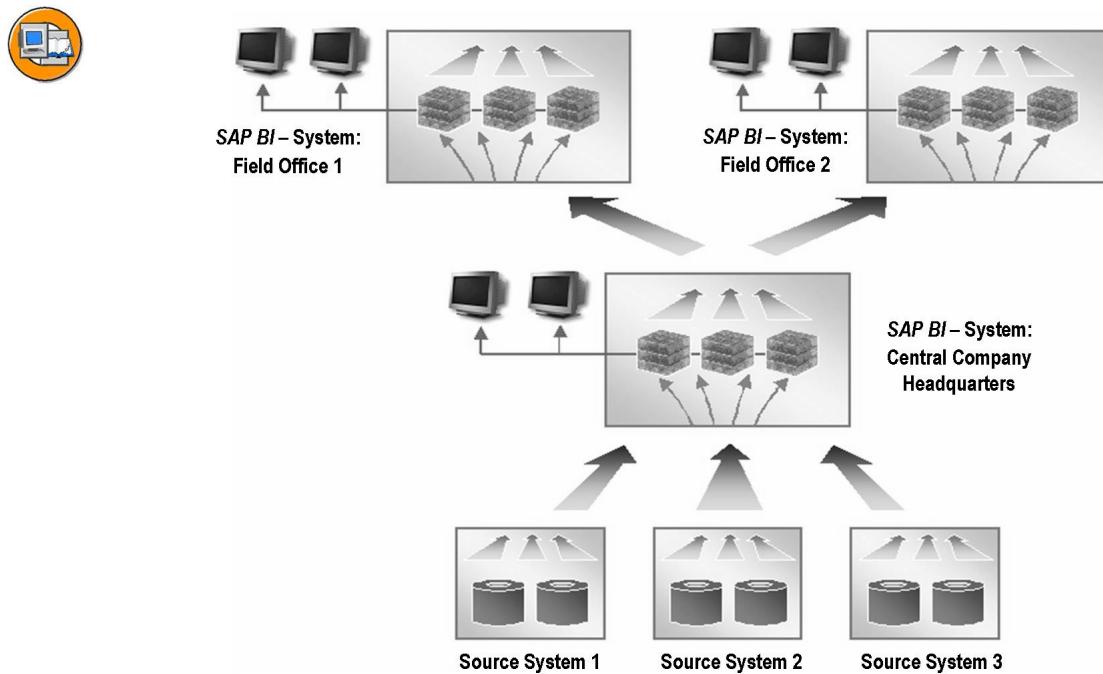


Figure 194: Replicating Data Mart Architecture

Aggregating Data Mart Architecture

With an *aggregating* architecture, also known as Outside-In architecture, data is updated from one or more BI systems to a central BI system.

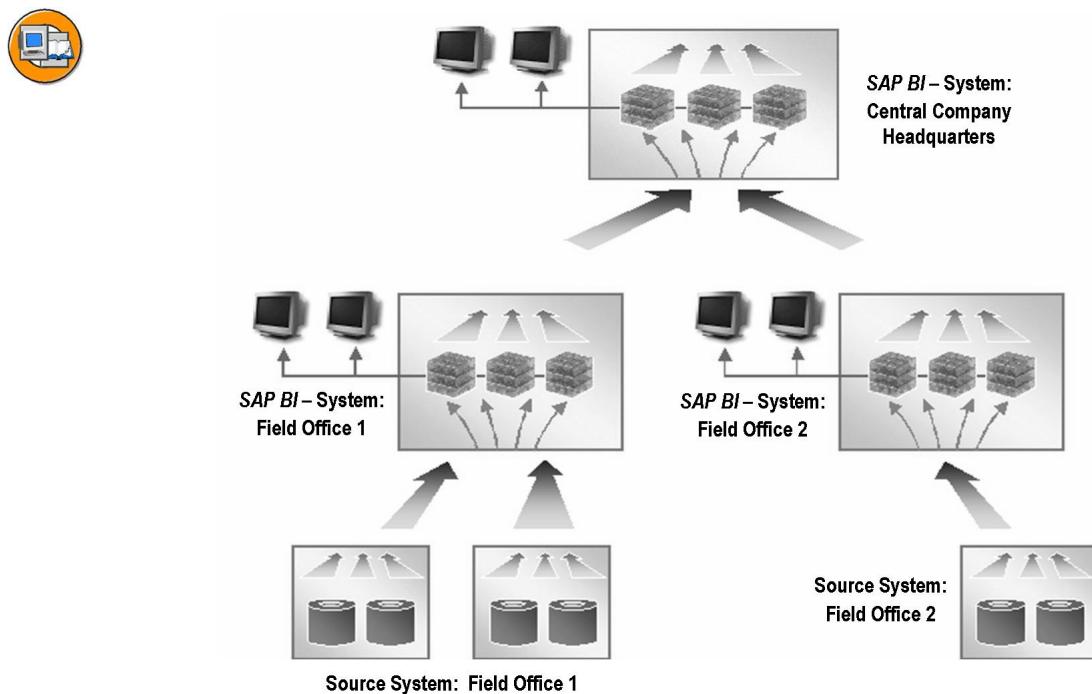


Figure 195: Aggregating Data Mart Architecture

Mixed Data Mart architecture

The following graphic illustrates a possible business scenario. Two BI systems in branch offices 1 and 2 serve as Data Marts. They transfer selected data to a central BI system. The central BI system transfers replicated data to the regional BI system.

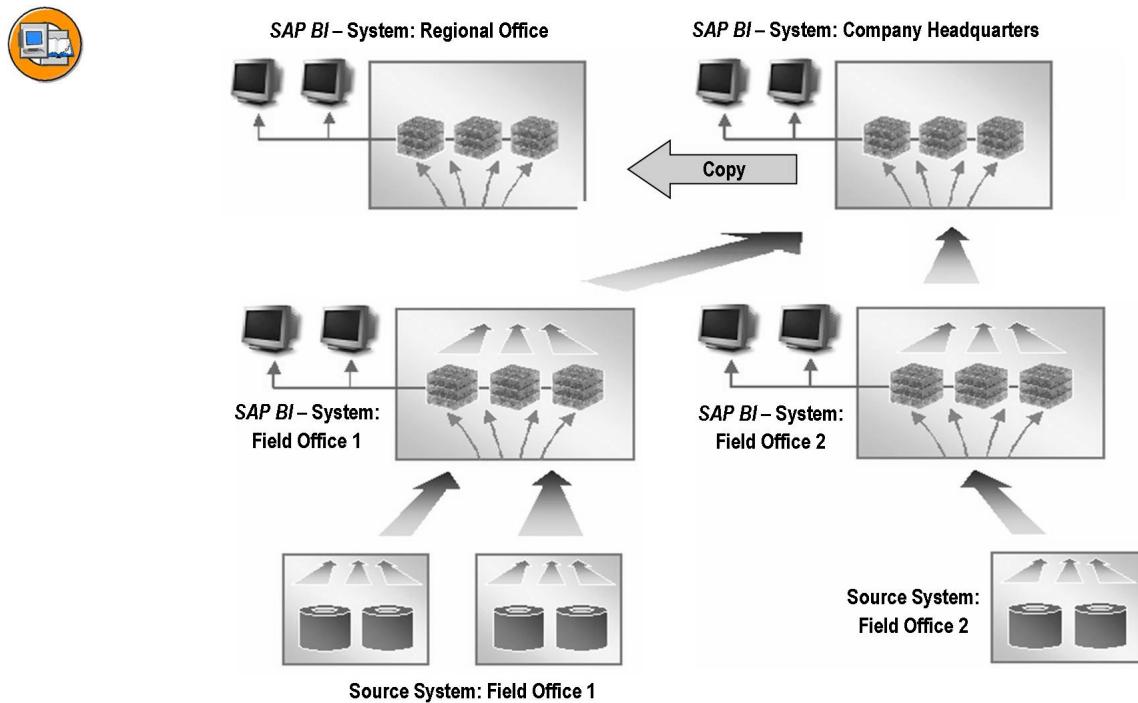


Figure 196: Mixed Data Mart Architecture

The Data Mart architectures described above illustrate the advantages of the Data Mart Interface:

- By adding several Data Marts to your system landscape, you can make it easier to set up and implement a data warehouse. Data Marts, which cover parts of the system landscape, can be implemented independently. It is easier to support and maintain each Data Mart than to do the same for very large systems.
- You can increase the speed with which data is transferred and the time required for reporting since the data formatting and staging processes are shorter.
- Data can be divided according to task areas and/or subsidiaries, and you can also perform analyses across the (aggregated) total datasets. For example, the company headquarters will need consolidated and selected data from the subsidiaries, whereas the subsidiaries only need specific data from the company headquarters.
- You can implement *hub and spoke* scenarios by placing a BI system at the center and integrating and homogenizing data from distributed systems.

Data Acquisition Using the Data Mart Interface

Data Mart Interface between BI systems

As mentioned above, the Data Mart Interface enables you to update data from one source BI system to one or more target BI systems. To do this, several steps are required.



1. Connect the *Source BI system* to the *Target BI system* with an RFC connection
2. Generate an Export DataSource in the *Source BI system*
3. Replicate the metadata of the Export DataSource to the *Target BI system*
4. Assign the Export DataSource to a Data Transfer Process
5. Maintain and Activate the Transformation Rules
6. Create and schedule a Process Chain to extract the data from the Export DataSource

Figure 197: Transferring Data Between BI Systems

To transfer data between BI systems, you must first link the source BI system to the target BI using an RFC connection.

When you create a source system, all BI systems behave in the same way as SAP systems, down to the source system type. The procedures, including all the settings, are the same as they are when you automatically create SAP systems.

In the second step, you generate an **Export DataSource** for the corresponding InfoProvider (InfoCube, DataStore object, characteristic InfoObject) in the source BI system with which the data and metadata can be transferred from a source BI system to a target BI system.

- *DataStore Object:*

The Export DataSource is automatically generated when you define DataStore objects.

- *InfoCube:*

In the InfoProvider tree for the *Data Warehousing Workbench*, choose the function *Generate Export DataSource* in the context menu for the InfoCube.

- *Characteristic InfoObject:*

In the InfoSource tree for the *Data Warehousing Workbench*, choose the function *Generate Export DataSource* in the context menu for the characteristic InfoObject (= InfoSource with direct update). Alternatively, in the maintenance screen for the characteristic InfoObject on the *Master Data/Texts* tab page, set the *Characteristic is Export DataSource* indicator. Depending on the settings you have made in the maintenance screen for the characteristic InfoObject, the system generates an Export DataSource for attributes, texts and hierarchies.

In the case of a InfoCube/DataStore object, the technical name of the Export DataSource consists of the prefix **8** and the technical name of the InfoCube/DataStore object:

8<technical name of the InfoCube/DataStore object>

The technical name of an Export DataSource for a characteristic InfoObject consists of the prefix **8**, the technical name of the characteristic InfoObject and the suffix M, T or H:

Export DataSource for attributes: 8<technical name of the characteristic InfoObject>M

Export DataSource for texts: 8<technical name of the characteristic InfoObject>T

Export DataSource for hierarchies: 8<technical name of the characteristic InfoObject>H

When the Export DataSource is generated, an **Extract Structure** is generated too, based on the InfoObjects of the corresponding InfoProvider. The technical name of the extraction structure is made up of the prefix **/BIC/CE** and the technical name of the Export DataSource.

/BIC/CE<technical name of the Export DataSource>

With InfoCubes and DataStore objects you can assign characteristics such as *Select* and *Hide* to fields in the Extract Structure. In the InfoProvider tree, choose the function *Maintain Export DataSource* in the context menu for the respective InfoProvider.

In the third step, replicate your Export DataSource. With this function in the target BI system, the metadata is transferred from the source BI system to the target BI system.

In the fourth step, you assign a Transformation to your Export DataSource. You can either accept the Transformation proposed by the system or specify your own Transformation.

In the fifth step, you maintain and activate transformation rules.

In the sixth step, you schedule a data transfer.

Exercise 13: Data Mart Interface

Exercise Objectives

After completing this exercise, you will be able to:

- Use an DataStore object for delta harmonizing of InfoCube data

Business Example

You want to evaluate the sales data from an external system using a InfoCube in BI. For delta harmonization you need to use an DataStore that will preprocess and subsequently be used to supply the InfoCube, however. The external data is provided using CSV files.

Task 1:

Sales data from two CSV files were loaded directly (using a DataSource and Data Transfer Process) to their InfoCube with the technical name **DM_CUBE##** and the description **BW350 DM InfoCube GR##**. The two CSV files have provided the following data:

BW350_NEW.csv (“new data”)

DOC_NUM	DOC_ITEM	MATERIAL	QUAN-TITY	UNIT	YEAR
RW-4711	10	M-01	100.00	L	2004

BW350_CHANGE.csv (“changed data”)

DOC_NUM	DOC_ITEM	MATERIAL	QUAN-TITY	UNIT	YEAR
RW-4711	10	M-01	80.00	L	2004

With the first upload, the data was loaded from the CSV file *BW350_NEW.csv* to the InfoCube. After the data was loaded, the quantity *100* was changed or adjusted to *80* in document number *RW-4711* in the external system. To update the data in the InfoCube, the changed data was loaded to the InfoCube with a second upload by means of the CSV file *BW350_CHANGE.csv*.

1. Display the data model for your InfoCube **DM_CUBE##** and check the following details:

Continued on next page

Dimensions	Characteristics (technical names)
Material	0DOC_ITM, 0DOC_NUM, 0MATERIAL
Time	0CALYEAR
Unit	0UNIT

Key figure (technical name)
0QUANTITY

Check the data in your InfoCube **DM_CUBE##** too.

2. Delete the data from your InfoCube in preparation for the following tasks.

Task 2:

You want to use your InfoCube to evaluate your external data. It is therefore necessary to use an DataStore object with the *Overwrite* update type for the key figure *Quantity* for delta harmonization. This means that your InfoCube has to be supplied by an DataStore object (Data Mart interface in the Myself system). First, you configure a data flow for updating the data from both CSV files with an DataStore object in your InfoCube. To do this, use the DataStore object with the technical name **DM_DS##** and the description **BW350 DM DS GR##** as well as the InfoSource with the technical name **BW350_DM_GR##** and the description **BW350 Data Mart GR##** to supply the DataStore object.

1. First, configure the data flow for your DataStore object **DM_DS##**. As you have already maintained the transfer rules, you only need to define update rules between your DataStore object **DM_DS##** and your InfoSource **BW350_DM_GR##**. Ignore the message *InfoObject 0RECORDMODE is missing in the InfoSource*. The system automatically generates a proposal for the update rules. The update type *Overwrite* is set for the key figure *Quantity* in the process.
2. Now create update rules between your InfoCube **DM_CUBE##** and your DataStore object **DM_DS##**.
3. Take a look at the structure of the data flow.

Continued on next page

Task 3:

After you have configured the data flow in task 2, load the sales data from the two CSV files to your InfoCube **DM_CUBE##** using your DataStore object **DM_DS##** and analyze yourCube again. First, load the “new” data from the CSV file *BW350_NEW* to your DataStore object and then to your InfoCube by init update. Load the “changed” data from the CSV file *BW350_CHANGE* to your DataStore object and then to your InfoCube by delta update.

1. In preparation for the next task you first need to download both CSV files to your workstation.
2. In the scheduler, create an InfoPackage with the description **GR## DS New Data** for the source system **I_EXTERN** for your InfoSource **BW350_DM_GR##**, to load the “new” data to your DataStore object **DM_DS##**. In the process, make the following settings in the scheduler:

Tab page *External Data*:

Field name	Value
<i>Load External Data</i>	Client Workstation
<i>File Type</i>	CSV File
<i>File Name</i>	N:\BW350_NEW.csv (→ using F4 help)
<i>Character for decimal point</i>	(Point).
<i>Number of headers to be ignored</i>	1

Tab page *Data Targets*:

<i>Select Data Targets</i>	<input checked="" type="checkbox"/>
<i>DataStore Object BW350 DM DS GR00</i>	<input checked="" type="checkbox"/>

Start your InfoPackage, check the processing in the monitor and display the data in your DataStore object **DM_DS##**. Enter the result in the following tables:

Table: *Active Data*

Continued on next page

DOC_- NUM	DOC_- ITEM	MATE- RIAL	QUAN- TITY	UNIT	CA- LYEAR

Table: *Change Log*

RQ	DP	RC	DOC_- NUM	DOC_- ITEM	MATERIAL
<ID1>					

QUANTITY	UNIT	CALYEAR	RM

RQ = REQUEST

<ID1> = Generic ID

DP = DATAPAKID

RC = RECORD

RM = RECORDMODE

3. Load the data from your DataStore object **DM_DS##** to your InfoCube **DM_CUBE##** by *init update*. Check the processing in the monitor and display the content of your InfoCube **DM_CUBE##**. Enter the result in the following table:

Result:

0DOC_- ITEM	0DOC_- NUM	0MATE- RIAL	0QUAN- TITY	0UNIT	0CA- LYEAR

Task 4:

You now want to load the “changed” data to your InfoCube using your DataStore object. To do this, create an InfoPackage with the description **GR## DS Changed Data** to load this data to your DataStore object **DM_DS##**. In the process, make the following settings in the scheduler:

1. Tab page *External Data*:

Continued on next page

Field name	Value
Load External Data	Client Workstation
File Type	CSV File
File Name	N:\BW350_CHANGE.csv (→ using the F4 help)
Character for decimal point	(Point).
Number of headers to be ignored	1

Tab page *Data Targets*:

Select Data Targets	✓
DataStore Object BW350 DM DS GR00	✓

Start your InfoPackage, check the processing in the monitor and display the data in your DataStore object **DM_DS##**. Enter the result in the following tables:

Table: *Active Data*

DOC_NUM	DOC_ITEM	MATERIAL	QUANTITY	UNIT	CALYEAR

Table: *Change Log*

RQ	DP	RC	DOC_NUM	DOC_ITEM	MATERIAL
<ID1>	1	1	RW-4711	10	M-01
<ID2>					
<ID3>					

QUANTITY	UNIT	CALYEAR	RM
100,000	L	2004	N
<ID2>			
<ID3>			

Continued on next page

Task 5:

1. Load the data from your DataStore object **DM_DS##** to your InfoCube **DM_CUBE##** by *delta update*. Check the processing in the monitor and display the content of your InfoCube **DM_CUBE##**. Enter the result in the following table:

Result:

0DOC-ITEM	0DOC-NUM	0MATERIAL	0QUANTITY	0UNIT	0CALYEAR
10	RW-4711	M-01	100,000	L	2004

Solution 13: Data Mart Interface

Task 1:

Sales data from two CSV files were loaded directly (using a DataSource and Data Transfer Process) to their InfoCube with the technical name **DM_CUBE##** and the description **BW350 DM InfoCube GR##**. The two CSV files have provided the following data:

BW350_NEW.csv (“new data”)

DOC_NUM	DOC_ITEM	MATERIAL	QUAN-TITY	UNIT	YEAR
RW-4711	10	M-01	100.00	L	2004

BW350_CHANGE.csv (“changed data”)

DOC_NUM	DOC_ITEM	MATERIAL	QUAN-TITY	UNIT	YEAR
RW-4711	10	M-01	80.00	L	2004

With the first upload, the data was loaded from the CSV file *BW350_NEW.csv* to the InfoCube. After the data was loaded, the quantity *100* was changed or adjusted to *80* in document number *RW-4711* in the external system. To update the data in the InfoCube, the changed data was loaded to the InfoCube with a second upload by means of the CSV file *BW350_CHANGE.csv*.

1. Display the data model for your InfoCube **DM_CUBE##** and check the following details:

Dimensions	Characteristics (technical names)
Material	0DOC_ITM, 0DOC_NUM, 0MATERIAL
Time	0CALYEAR
Unit	0UNIT

Key figure (technical name)
0QUANTITY

Continued on next page

Check the data in your InfoCube **DM_CUBE##** too.

- To display the data model, choose:

Data Warehousing Workbench: *Modeling* → *InfoProvider* → *InfoProvider Tree* → *InfoArea: BW Training* → *BW Customer Training* → *BW350 Extraction SAP Components* → *BW350 Group ## (T_BW350_GR##)* → *InfoCube: DM_CUBE##* → *Context Menu: Display Data Model*.

Choose to close the window on the right.

- To display the content of your InfoCube, select the following from the context menu:

Manage → *Tab Page: Contents* → *Pushbutton: InfoCube Content* → *Field Selection for Output*.

On the maintenance screen, mark the checkboxes for:

- BW: Document Line No
- BW: Document Number
- Material
- Calendar Year
- Unit of Measure

Choose *Execute* twice.

0DOC-ITEM	0DOC-NUM	0MATERIAL	0QUANTITY	0UNIT	0CALYEAR
10	RW-4711	M-01	100,00	L	2004
10	RW-4711	M-01	80,00	L	2004



Note: Depending upon the setting in your use profile in BI, you may be seeing either the comma or the period as the “decimal character.”

- Delete the data from your InfoCube in preparation for the following tasks.

- In the context menu for your InfoCube, choose: *Manage*

Choose the *Requests* tab page and select both requests.

Choose *Delete*, the select *Yes* when prompted and if necessary *Refresh*.

Continued on next page

Task 2:

You want to use your InfoCube to evaluate your external data. It is therefore necessary to use an DataStore object with the *Overwrite* update type for the key figure *Quantity* for delta harmonization. This means that your InfoCube has to be supplied by an DataStore object (Data Mart interface in the Myself system). First, you configure a data flow for updating the data from both CSV files with an DataStore object in your InfoCube. To do this, use the DataStore object with the technical name **DM_DS##** and the description **BW350 DM DS GR##** as well as the InfoSource with the technical name **BW350_DM_GR##** and the description **BW350 Data Mart GR##** to supply the DataStore object.

1. First, configure the data flow for your DataStore object **DM_DS##**. As you have already maintained the transfer rules, you only need to define update rules between your DataStore object **DM_DS##** and your InfoSource **BW350_DM_GR##**. Ignore the message *InfoObject 0RECORDMODE is missing in the InfoSource*. The system automatically generates a proposal for the update rules. The update type *Overwrite* is set for the key figure *Quantity* in the process.
 - a) In the context menu for your DataStore object **DM_DS##** choose the function *Create Update Rules*.

Choose the following value for the InfoSource:

Field name	Value
<i>InfoSource</i>	BW350_DM_GR##

The maintenance screen *Create Update Rules: Initial Screen* opens.

Choose *Continue*.

Choose *Continue* again. By doing this, you confirm that the InfoObject *0RECORDMODE* is not part of the InfoSource.

The maintenance screen *Create Update Rules: Rules* opens. Choose *Activate* to accept the proposed update rules.

Choose *Back*

Continued on next page

2. Now create update rules between your InfoCube **DM_CUBE##** and your DataStore object **DM_DS##**.

- a) In the context menu for your InfoCube **DM_CUBE##** choose the function *Create Update Rules*.

Choose the following value for the data source:

Field name	Value
<i>DataStore Object</i>	DM_DS##

The maintenance screen *Create Update Rules: Initial Screen* opens.

Choose *Continue*.

The maintenance screen *Create Update Rules: Rules* opens.

Choose *Activate* to accept the proposed update rules.

Choose *Back*

3. Take a look at the structure of the data flow.

- a) In the context menu for your InfoCube **DM_CUBE##** choose the function *Display Data Flow*.

Choose *Back*

Task 3:

After you have configured the data flow in task 2, load the sales data from the two CSV files to your InfoCube **DM_CUBE##** using your DataStore object **DM_DS##** and analyze yourCube again. First, load the “new” data from the CSV file *BW350_NEW* to your DataStore object and then to your InfoCube by init update. Load the “changed” data from the CSV file *BW350_CHANGE* to your DataStore object and then to your InfoCube by delta update.

1. In preparation for the next task you first need to download both CSV files to your workstation.
- a) *Transaction /OSO04 (= Business Workplace) → Folder: Material for BW Training → Double-click → Folder: BW350 Files for BW Training → Double-click → BW350 Files for Training → Double-click → Tab Page: Attachments → Context Menu for BW350_NEW: Export Attachment:*

Choose the **N:** folder and save the data under the name **BW350_NEW**. Follow the same procedure for the second file *BW350_CHANGE*.

You can display both files with *Start → Programs → Notepad*.

Continued on next page

2. In the scheduler, create an InfoPackage with the description **GR## DS New Data** for the source system **I_EXTERN** for your InfoSource **BW350_DM_GR##**, to load the “new” data to your DataStore object **DM_DS##**. In the process, make the following settings in the scheduler:

Tab page *External Data*:

Field name	Value
<i>Load External Data</i>	Client Workstation
<i>File Type</i>	CSV File
<i>File Name</i>	N:\BW350_NEW.csv (→ using F4 help)
<i>Character for decimal point</i>	(Point).
<i>Number of headers to be ignored</i>	1

Tab page *Data Targets*:

<i>Select Data Targets</i>	<input checked="" type="checkbox"/>
<i>DataStore Object BW350 DM DS GR00</i>	<input checked="" type="checkbox"/>

Start your InfoPackage, check the processing in the monitor and display the data in your DataStore object **DM_DS##**. Enter the result in the following tables:

Table: *Active Data*

DOC_- NUM	DOC_- ITEM	MATE- RIAL	QUAN- TITY	UNIT	CA- LYEAR

Table: *Change Log*

RQ	DP	RC	DOC_- NUM	DOC_- ITEM	MATERIAL
<ID1>					

QUANTITY	UNIT	CALYEAR	RM

RQ = REQUEST

Continued on next page

<ID1> = Generic ID

DP = DATAPAKID

RC = RECORD

RM = RECORDMODE

- a) Data Warehousing Workbench: *InfoSources* → *InfoSource Tree* → *Application Component: BW Training* → *BW350 SAP Component Extraction* → *Group## (ZT_BW350_##)* → *InfoSource: BW350 Data Mart GR## (BW350_DM_GR##)* → *Source System: I_EXTERN* → *Context Menu: Create InfoPackage*

Enter the following values:

Field name	Value
<i>InfoPackage Description</i>	GR## DS New data

Choose *Save (Enter)*.

The maintenance screen *Scheduler (Maintain InfoPackage)* opens.

Choose the *External Data* tab page and enter the following values:

Field name	Value
<i>Load External Data</i>	Client Workstation
<i>File Type</i>	CSV File
<i>File Name</i>	N:\BW350_NEW.csv (→ using F4 help)
<i>Character for decimal point</i>	(Point).
<i>Number of headers to be ignored</i>	1

Choose the *Data Target* tab page and enter the following values:

<i>Select Data Targets</i>	<input checked="" type="checkbox"/>
<i>DataStore Object BW350 DM DS GR00</i>	<input checked="" type="checkbox"/>

On the *Schedule* tab page, choose: *Start*, then *Monitor*.

Choose the *Detail* tab page and *Refresh* if necessary.

Continued on next page

- b) To display the data in your DataStore object **DM_DS##**, choose  *Manage Data Target....*

On the *Content* tab page, choose the *Activation Queue (= New Data)* pushbutton or *Active Data* or *Change Log*:

Table: *Activation Queue (= New Data)*

This table contains no data, since the *Activate Data for DataStore Object Automatically* indicator is set in the DataStore object definition.

Table: *Active Data*

DOC_- NUM	DOC_- ITEM	MATE- RIAL	QUAN- TITY	UNIT	CA- LYEAR
RW-4711	10	M-01	100,000	L	2004

Table: *Change Log*

RQ	DP	RC	DOC_- NUM	DOC_- ITEM	MATE- RIAL
<ID1>	1	1	RW-4711	10	M-01

QUANTITY	UNIT	CALYEAR	RM
100,000	L	2004	N

N = New Image

3. Load the data from your DataStore object **DM_DS##** to your InfoCube **DM_CUBE##** by *init update*. Check the processing in the monitor and display the content of your InfoCube **DM_CUBE##**. Enter the result in the following table:

Result:

Continued on next page

0DOC_-ITEM	0DOC_-NUM	0MATERIAL	0QUANTITY	0UNIT	0CALENDAR-YEAR

- a) In the context menu for your DataStore object **DM_DS##**, choose the function *Update 3.x Data in Data Target*.

The *Update DataStore Table* dialog box opens.

Choose the *init update* function. (With the init and full update, the *Active Data* table is read!)

The maintenance screen *Scheduler (Maintain InfoPackage)* opens.

Choose the *Data Targets* tab page and make the following settings:

Select Data Targets	<input checked="" type="checkbox"/>
BW350 DM InfoCube GR00	<input checked="" type="checkbox"/>

On the *Schedule* tab page, choose *Start*, then *Monitor*.

Choose the *Detail* tab page and *Refresh* if necessary.

- b) To display the data in your Basic Cube **DM_CUBE##**, choose *Manage Data Target....*

Tab Page: Contents → Pushbutton: InfoCube Content → Field Selection for Output.

On the maintenance screen, mark the checkboxes for:

- BW: Document Line No.
- BW: Document Number
- Material
- Calendar Year
- Unit of Measure

Choose *Execute* twice.

0DOC_-ITEM	0DOC_-NUM	0MATERIAL	0QUANTITY	0UNIT	0CALENDAR-YEAR
10	RW-4711	M-01	100,000	L	2004

Continued on next page

Task 4:

You now want to load the “changed” data to your InfoCube using your DataStore object. To do this, create an InfoPackage with the description **GR## DS Changed Data** to load this data to your DataStore object **DM_DS##**. In the process, make the following settings in the scheduler:

1. Tab page *External Data*:

Field name	Value
<i>Load External Data</i>	Client Workstation
<i>File Type</i>	CSV File
<i>File Name</i>	N:\BW350_CHANGE.csv (→ using the F4 help)
<i>Character for decimal point</i>	(Point).
<i>Number of headers to be ignored</i>	1

Tab page *Data Targets*:

<i>Select Data Targets</i>	<input checked="" type="checkbox"/>
<i>DataStore Object BW350 DM DS GR00</i>	<input checked="" type="checkbox"/>

Start your InfoPackage, check the processing in the monitor and display the data in your DataStore object **DM_DS##**. Enter the result in the following tables:

Table: *Active Data*

DOC_- NUM	DOC_- ITEM	MATE- RIAL	QUAN- TITY	UNIT	CA- LYEAR

Table: *Change Log*

RQ	DP	RC	DOC_- NUM	DOC_- ITEM	MATERIAL
<ID1>	1	1	RW-4711	10	M-01
<ID2>					
<ID3>					

Continued on next page

QUANTITY	UNIT	CALYEAR	RM
100,000	L	2004	N
<ID2>			
<ID3>			

- a) Data Warehousing Workbench: *InfoSources* → *InfoSource Tree* → *Application Component*: *BW Training* → *BW350 SAP Component Extraction* → *Group## (ZT_BW350_##)* → *InfoSource*: *BW350 Data Mart GR## (BW350_DM_GR##)* → *Source System*: *I_EXTERN* → *Context Menu*: *Create InfoPackage*

Enter the following values:

Field name	Value
<i>InfoPackage name</i>	GR## DS Changed data

Choose *Save (Enter)*.

The maintenance screen *Scheduler (Maintain InfoPackage)* opens.

Choose the *External Data* tab page and enter the following values:

Field name	Value
<i>Load External Data</i>	Client Workstation
<i>File Type</i>	CSV File
<i>File Name</i>	N:\BW350_CHANGE.csv (→ using the F4 help)
<i>Character for decimal point</i>	(Point).
<i>Number of headers to be ignored</i>	1

Choose the *Data Target* tab page and enter the following values:

<i>Select Data Targets</i>	<input checked="" type="checkbox"/>
<i>DataStore Object BW350 DM DS GR00</i>	<input checked="" type="checkbox"/>

On the *Schedule* tab page, choose: *Start*, then *Monitor*.

Choose the *Detail* tab page and *Refresh* if necessary.

Continued on next page

- b) To display the data in your DataStore object **DM_DS##**, choose  *Manage Data Target....*

On the *Content* tab page, choose the *Activation Queue* (= *New Data*) pushbutton or *Active Data* or *Change Log*:

Table: *Activation Queue* (= *New Data*)

This table contains no data, since the *Activate Data for DataStore Object Automatically* indicator is set in the DataStore object definition.

Table: *Active Data*

DOC_- NUM	DOC_- ITEM	MATE- RIAL	QUAN- TITY	UNIT	CA- LYEAR
RW-4711	10	M-01	80,000	L	2004

If you compare this result with the result in subtask 2 you can see that the values in the section for “new” data have been overwritten by the “changed” data. In particular, the quantity 100 was overwritten by the quantity 80.

Table: *Change Log*

RQ	DP	RC	DOC_- NUM	DOC_- ITEM	MATE- RIAL
<ID1>	1	1	RW-4711	10	M-01
<ID2>	1	1	RW-4711	10	M-01
<ID3>	1	1	RW-4711	10	M-01

QUANTITY	UNIT	CALYEAR	RM
100,000	L	2004	N
100,000-	L	2004	X
80,000	L	2004	

If you compare this result with the result in subtask 2, you can see that an additional request <ID2> was generated containing a before image (= X) and an after image (= blank) for the first data record.

Continued on next page

Task 5:

- Load the data from your DataStore object **DM_DS##** to your InfoCube **DM_CUBE##** by *delta update*. Check the processing in the monitor and display the content of your InfoCube **DM_CUBE##**. Enter the result in the following table:

Result:

0DOC-ITEM	0DOC-NUM	0MATERIAL	0QUANTITY	0UNIT	0CALENDAR
10	RW-4711	M-01	100,000	L	2004

- In the context menu for your DataStore object **DM_DS##**, choose the function *Update 3.x Data in Data Target*.

The *Update DataStore Table* dialog box opens.

Choose the *delta update* function. (With the delta update, the *Change Log* table is read!)

The maintenance screen *Scheduler (Maintain InfoPackage)* opens.

Choose the *Data Targets* tab page and make the following settings:

<i>Select Data Targets</i>	<input checked="" type="checkbox"/>
<i>BW350 DM InfoCube GR00</i>	<input checked="" type="checkbox"/>

On the *Schedule* tab page, choose *Start*, then *Monitor*.

Choose the *Detail* tab page and *Refresh* if necessary.

- To display the data in your Basic Cube **DM_CUBE##**, choose *Manage Data Target....*

Tab Page: *Contents* → Pushbutton: *InfoCube Content* → *Field Selection for Output*.

On the maintenance screen, mark the checkboxes for:

- BW: Document Line No.
- BW: Document Number
- Material
- Calendar Year
- Unit of Measure

Continued on next page

Choose  Execute twice.

0DOC-ITEM	0DOC-NUM	0MATERIAL	0QUANTITY	0UNIT	0CALYEAR
10	RW-4711	M-01	100,000	L	2004
10	RW-4711	M-01	-20,000	L	2004

1. Data record: Init update from the DataStore object.
2. Data record: Delta update from the DataStore object.



Lesson Summary

You should now be able to:

- Extract the data from a source BI system to another target BI system



Unit Summary

You should now be able to:

- Extract the data from a source BI system to another target BI system

Unit 11

Optional: Application-Specific Data Acquisition

Unit Overview

This unit includes information about acquiring application-specific data.



Unit Objectives

After completing this unit, you will be able to:

- Design and define DataSources in CO-PA
- Build an InfoCube and maintain transformation rules
- Manage CO-PA extraction
- Use CO-PA tools for BI
- Position FI-SL in the SAP system
- Describe FI-SL data structures
- Generate FI-SL DataSources
- Explain the particular methods used to extract data from FI-SL
- Create generic DataSources for set hierarchies

Unit Contents

Lesson: Data Acquisition from Profitability Analysis (Optional).....	488
Lesson: Data Acquisition from Special Ledger (Optional).....	512

Lesson: Data Acquisition from Profitability Analysis (Optional)

Lesson Overview

This lesson explains how data is acquired from financial areas such as Profitability Analysis (CO-PA).



Lesson Objectives

After completing this lesson, you will be able to:

- Design and define DataSources in CO-PA
- Build an InfoCube and maintain transformation rules
- Manage CO-PA extraction
- Use CO-PA tools for BI

Business Example

The ABC Company has implemented Profitability Analysis (CO-PA) and wants to use this data in the SAP Business Information Warehouse to compile reports for management. Management requires analyses of gross and net sales figures from the individual sales organizations.

You are responsible for extracting the data from the SAP OLTP system and importing it into BI. To do so, you have to create the necessary components in the OLTP and BI systems.

Using Available Accounting Data

The Profitability Analysis (CO-PA) solution generates data that is critical for preparing management reports.

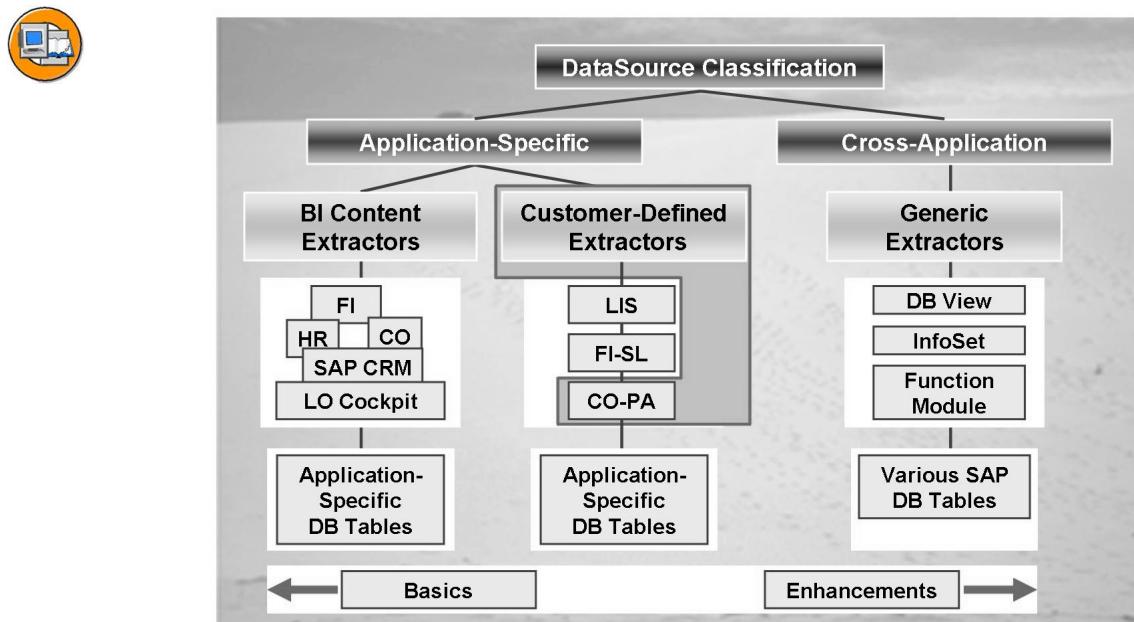


Figure 198: Extracting Data from CO-PA: Overview Diagram

CO-PA collects all of the OLTP data for calculating contribution margins (sales, cost of sales, overhead costs)-

CO-PA also has powerful reporting tools and planning functions.

The CO-PA reporting facility is limited in two respects, however:

- The integrated cross-application reporting concept is not as differentiated as it is in BI.
- The OLTP system is optimized for transaction processing, and a high reporting workload would have a negative impact on the overall performance of the system.

Using BI as a reporting solution for CO-PA eliminates these problems

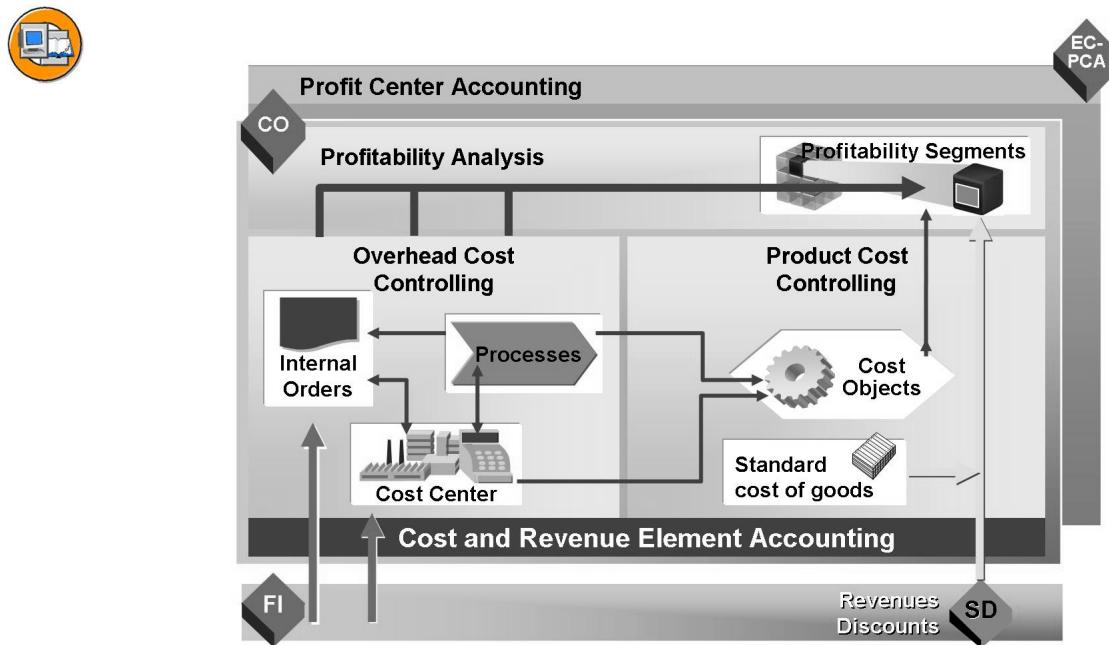


Figure 199: Flow of Actual Values: Overview

During billing in SD, revenues and discounts are transferred to profitability segments in Profitability Analysis. At the same time, sales quantities are valued using the standard cost of goods manufactured, as specified in the cost component split from CO-PC.

In Overhead Cost Controlling, primary postings are made to objects in Overhead Cost Controlling and assigned to the relevant cost object on a source-related basis. The actual cost of goods manufactured is also assigned to the cost object. At the same time, the performing cost centers are credited. From the point of view of profitability analysis, this leads to under- or over-absorption of the performing cost centers and to production variances for the cost objects involved (such as production orders).

The production variances calculated for the cost objects (in this case, production orders), in other words, the difference between the actual cost of goods manufactured and the standard costs, are divided into variance categories and settled to profitability segments. The overhead costs that remain for the objects in Overhead Cost Controlling are assigned to the source profitability segments.

What are the top products and who are the top customers in our various divisions and areas? This is just one of the typical questions that can be answered with Profitability Analysis (CO-PA).

The wide variety of analysis and planning functions in CO-PA allow you to plan, monitor, and control the success of your company in terms of product-oriented, customer-oriented and organizational dimensions of multidimensional profitability segments.

Data Sources

Profitability Analysis (CO-PA) provides the following data:

- characteristics
- characteristic values
- profitability segment
- key figures



Characteristics

- Characteristics are levels about which information is required.
Example: Divisions, regions, customer groups

Characteristic values

- Characteristic values are values that can be assumed by a characteristic.
Example for the characteristic 'region': Eastern region, northern region

Profitability segment

- A profitability segment is a combination of available characteristic values.
■ Example: Computer division, eastern region

Figure 200: Basic Concepts

Characteristics are the fields in an operating concern according to which the data can be differentiated in Profitability Analysis.

Each characteristic in an operating concern has a series of valid characteristic values.

A profitability segment is a fixed combination of valid characteristic values.

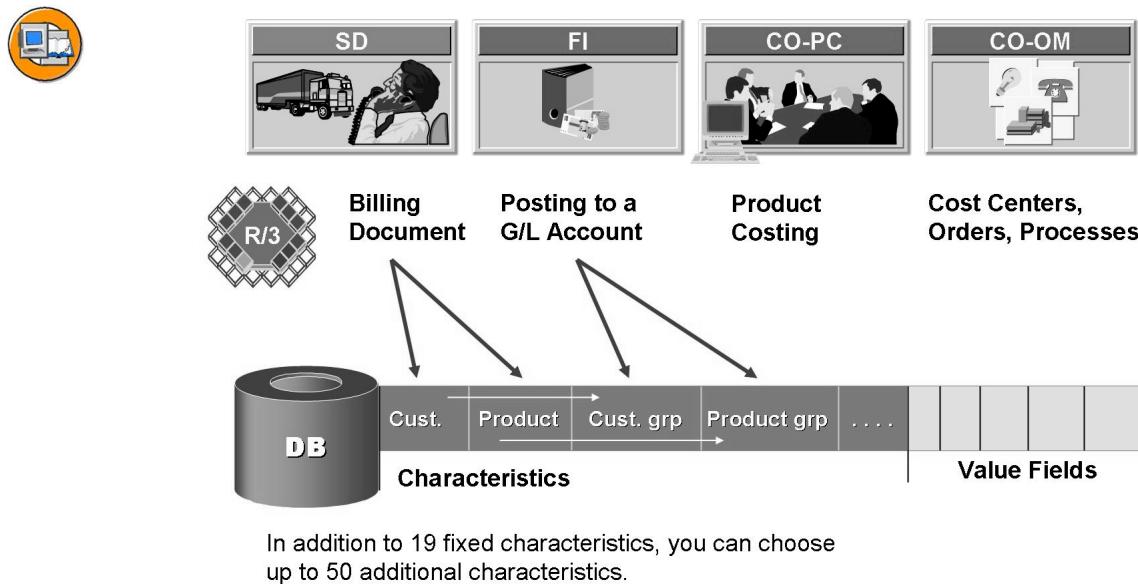


Figure 201: Characteristics

Some characteristics are predefined in each operating concern. This includes customer, material, company code, and others. You can call up a full list of the fixed characteristics by displaying the data structures in the operating concern.

In addition to these fixed characteristics, you can define up to 50 characteristics of your own. In most cases, you will be able to satisfy your profitability analysis requirements with between 10 and 20 characteristics.

Descriptive attributes (customer, material, time) are stored as characteristics so that the data can be analyzed according to several dimensions.

In addition to the independent characteristics that can be found on a sales document (customer, material, fiscal period, sales area, and so on) several characteristics can be derived from these (customer group, material hierarchy, sales hierarchy). When costs are posted from CO-OM, the most detailed characteristics are usually initial values (blank or zero) since the costs can only be properly assigned to objects that are less detailed. Marketing costs, for example, could be correctly assigned to the customer group and the article.

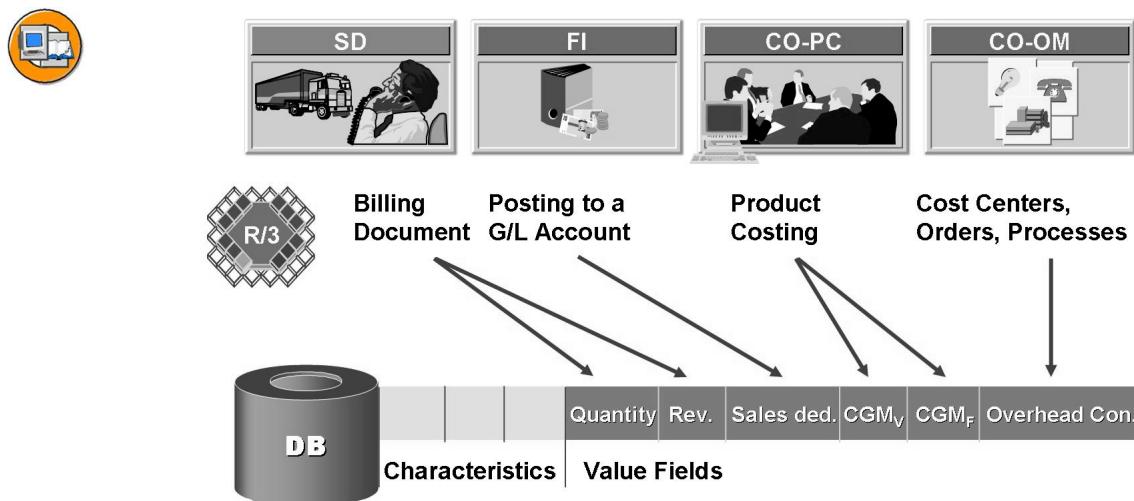


Figure 202: Value Fields

Key figures (revenue, cost of goods sold, overhead costs) are stored in value fields to make the contribution margin more transparent. Depending on the data source, some value fields are equal to zero, while others are not. In a sales document, for example, the sales quantity, revenue, rebate, cost of goods sold, (calculated from the product costs in CO-PC) and any accruals are not equal to zero. When costs are posted from CO-OM, however, all of these value fields are equal to zero. Other fields that are used to record different cost elements (marketing costs, for example) are not equal to zero in these data records.

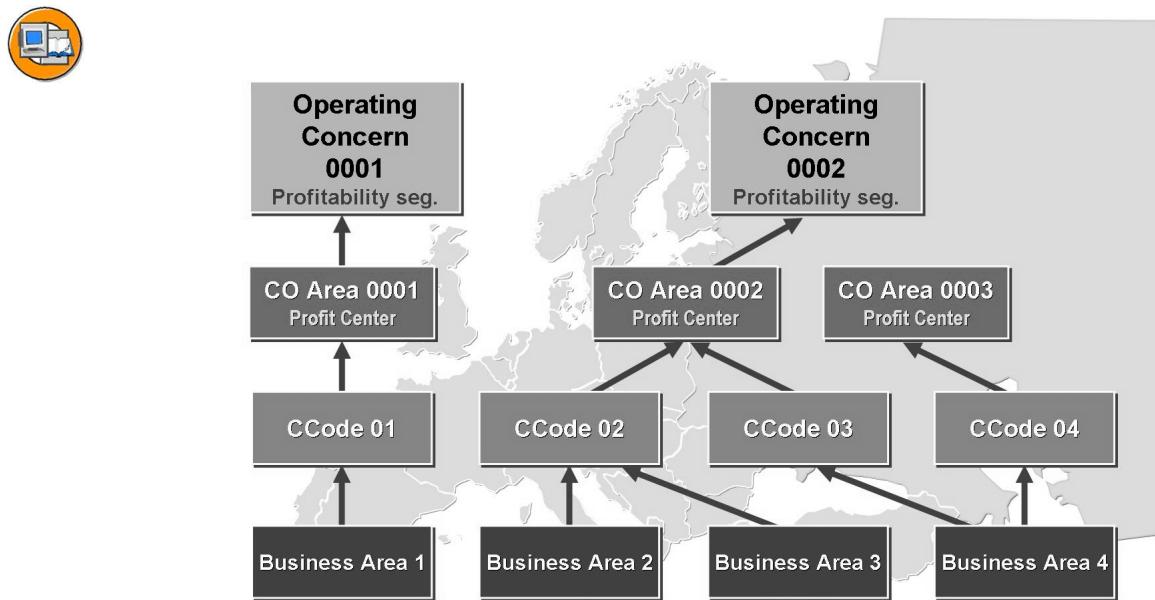


Figure 203: Organizational Structure

The value fields and characteristics that are required to conduct detailed analyses of the contribution margin vary considerably both from industry to industry and between individual customers. In CO-PA, therefore, you can configure the structure of one or more operating concerns in each individual installation. An operating concern is an organizational structure that groups controlling areas together in the same way as controlling areas group companies together. There is usually only one operating concern for each installation.

Since value fields and characteristics can be defined individually in each customer installation, it is not possible to ship all of the required data structures (and the programs for accessing these structures) with the R/3 installation CD. Instead, these structures have to be generated when CO-PA is configured (similar to an InfoCube in BI).

Data Structures

Accounting data is stored in different database tables.

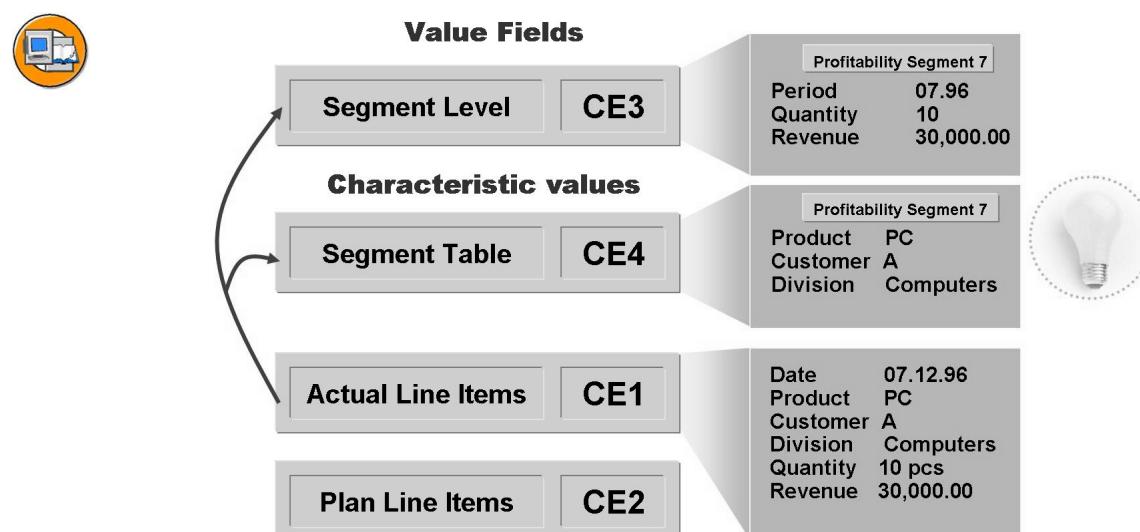


Figure 204: Database Structures in CO-PA

Line items are stored in separate tables: CE1xxxx (xxxx is the name of the operating concern) contains the actual line items and CE2xxx the plan line items. Note that these tables contain the value fields and characteristics and are therefore generated tables.

Line items contain some information at document level (CO-PA document number, sales document number, posting date) that is mostly too detailed for analysis purposes. CO-PA maintains an initial summarization of the data used by all CO-PA functions (reporting, planning, assessments, settlements, realignments, and so on.)

The characteristics that describe the market are first separated from the rest of the line items. Each combination of characteristic values is coded in a profitability segment number. The link between the profitability segment number and characteristic values is maintained in a separate table - the segment table CE4xxxx. Certain characteristics can be excluded from this process. These are then stored in the line items only and not in the segment table. These characteristics can only be analyzed to a limited extent. Characteristics that are differentiated to a large extent (such as the customer order number) are usually excluded to reduce the volume of data.

The value fields are summarized at the profitability segment and period levels (as well as other characteristics at the business transaction level: plan/actual indicator, record type, and plan version) and stored together with these fields in a second table known as the segment level CE3xxxx. This table contains the total values of the period for each profitability segment number.

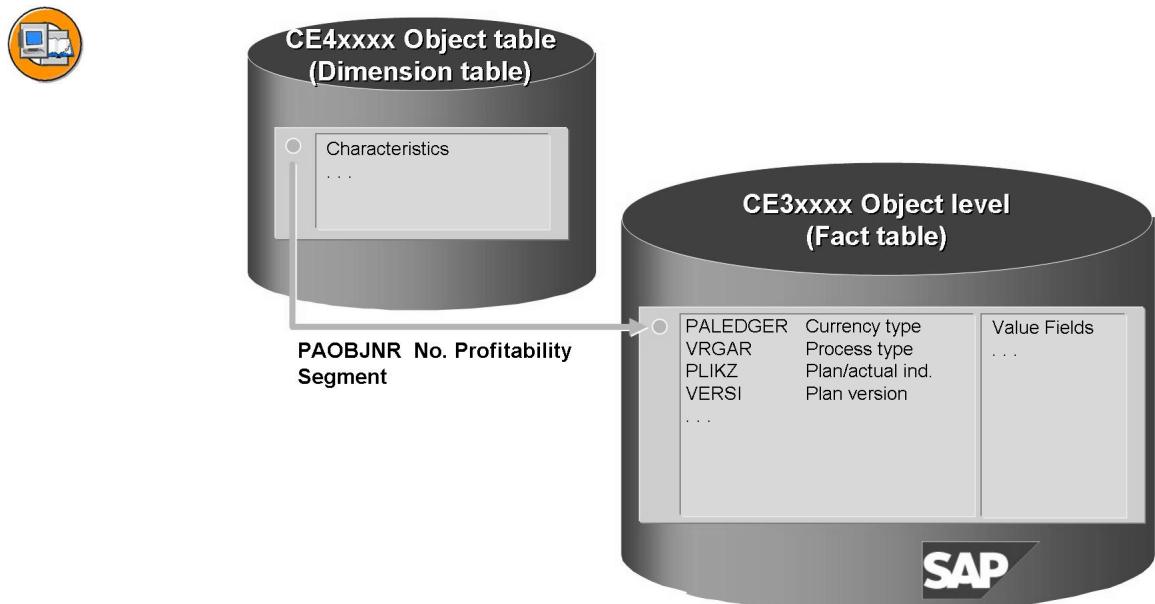


Figure 205: Storage Procedure Architecture

You can compare an operating concern (represented by the associated segment table and segment level) with an InfoCube. The InfoCube comprises a dimension table (the segment table) and a fact table (the segment level). Unlike the fact table of an InfoCube, the segment level key contains other keys (the record type, for example) in addition to the key field from the segment table.

Characteristics in CO-PA correspond to the characteristics (or attributes) in InfoCubes; the value fields can be regarded as key figures with an additional summarization in each characteristic.

Summarization levels for an operating concern have the same function as aggregates for an InfoCube. The difference is that aggregates for InfoCubes are managed together with the InfoCube itself (meaning that in a roll-up, all the aggregates contain the same figures as the InfoCube they are based on) while summarization levels are updated at regular intervals (usually every day).

Line items in CO-PA can be compared with the line items in the DataStore Object. These are also comparable with line items in the communication structure directly before they are posted to an InfoCube.

Managing Data Acquisition

This section describes what happens during data acquisition.

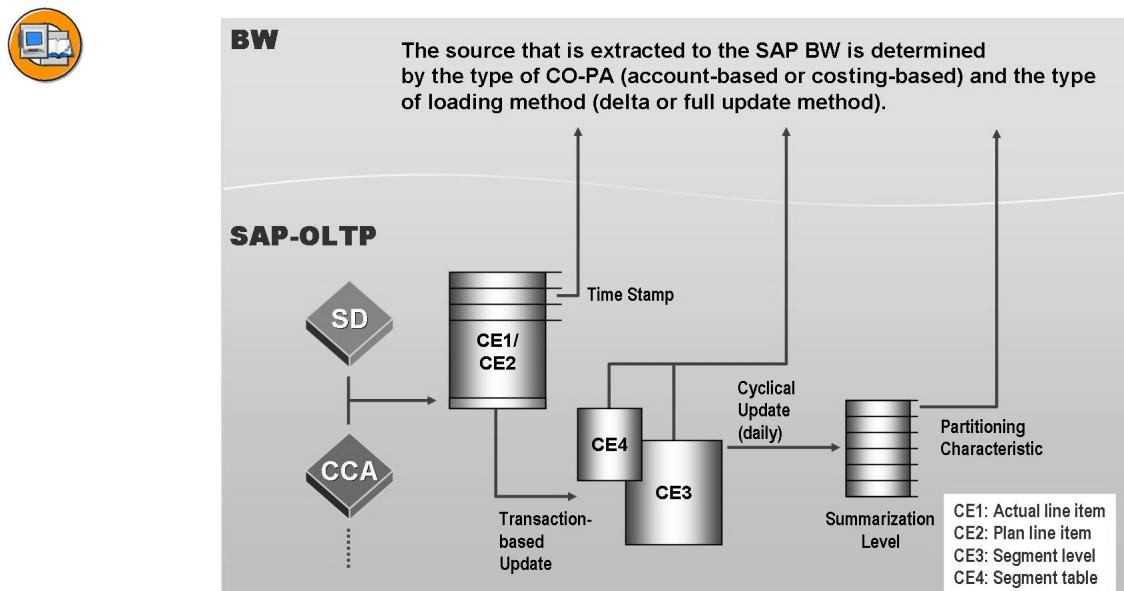


Figure 206: Update Model: Overview

To speed up access to data, CO-PA can also maintain summarized copies of segment tables and segment levels known as **summarization levels**. Each summarization level contains all of the value fields and a subset of the characteristics. To ensure that a correct aggregate is formed, some detailed characteristics are excluded in each summarization level. If a characteristic X (customer number, for example) is contained in a summarization level, all of the characteristics derived from X (customer group, for example) should also be contained in the same summarization level. Summarization levels are updated from time to time in a mass update run and not each time a line item is posted. The update procedure is controlled by time stamp information that is stored in each line item and in the summarization level catalog. You can use the time stamp to determine at any time whether a certain line item has already been entered in a specific summarization level.

In BI, you define an InfoPackage for the initial uploading process and further InfoPackages for a delta update from a CO-PA InfoSource. When an InfoPackage is run, the BI Staging Engine calls the CO-PA transaction data interface. Once you have generated an InfoSource, you need to start the initial loading process. After this you can run delta updates. Note that the CO-PA extraction program for BI uses the same replication method as the update program for CO-PA summarization levels for updating summarization levels. On the BI side, only data that is at least 30 minutes old is received. SAP Note 392876 contains further information about this **safety delta**.

To provide DataSources for BI, all CO-PA DataSources (and the operating concern itself) must be generated in the source system.

This is done in the IMG for SAP OLTP (transaction SPRO).

A CO-PA DataSource can be defined at the operating concern and client levels.

The DataSource consists of the following information:

- Name of the operating concern
- Client
- Data source for full updates (summarization level, segment level, line item)
- Subset of the characteristics to determine the level of detail that is replicated
- Subset of the value fields
- Update status (InfoSource defined, full update running, delta update running, update completed, invalid)
- Time stamp used to determine which data has already been loaded into BI

CO-PA in the BI IMG of the SAP OLTP System (*transaction SBIW → Settings for application specific data sources → Profitability Analysis*):

- **Procedure for setting up the replication model**

This chapter provides the information needed at the beginning of the installation scenario.

- **Assigning key figures**

In this step you assign elements of an accounting schema to some of the key figures that are predefined in CO-PA. You can add these fields to DataSources.

- **Creating the DataSource**

Create a DataSource to extract transaction data from costing-based or account-based CO-PA.

- **Tools: Displaying detailed information about the DataSource**

Display details about the DataSource such as header information, information about the delta process, and the field list.

- **Tools: Simulating delta process initialization**

Expert tool for testing existing CO-PA systems (with a high volume of data). Transaction KEB5 in OLTP.

- **Tool: Activating debugging support**

This transaction should only be used by SAP Support.

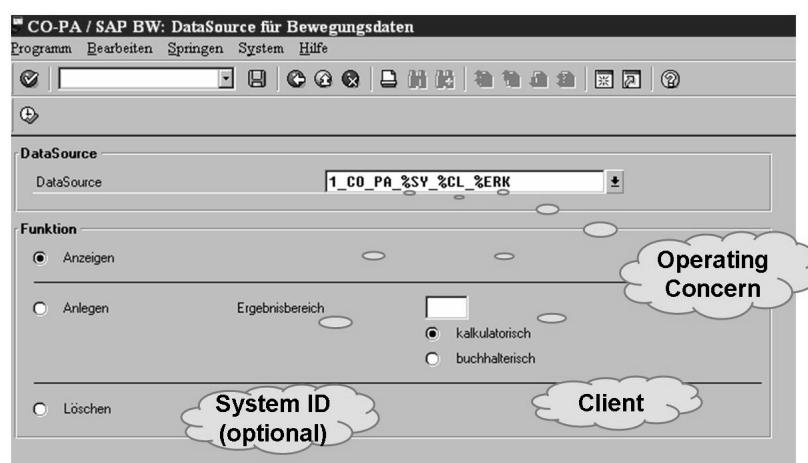


Figure 207: Introduction: Creating DataSources

Since a DataSource is always defined at the operating concern and client levels, a standard name is always generated, for example 1_CO_PA_<%CL>_<%ERK>. Additionally <%SYS> = optional system ID. You can change this name if necessary. The prefix 1_CO_PA is mandatory, however. We recommend that you only extend the standard name.

- **Example:** In the SAP System with ID P31 and client 100, the standard name of the DataSource for the IDES operating concern is: 1_CO_PA_P31_100_IDES.

Modify the name of the DataSource, if necessary, to distinguish it from other DataSources. You can create a new DataSource with this name using the relevant button on the initial screen.

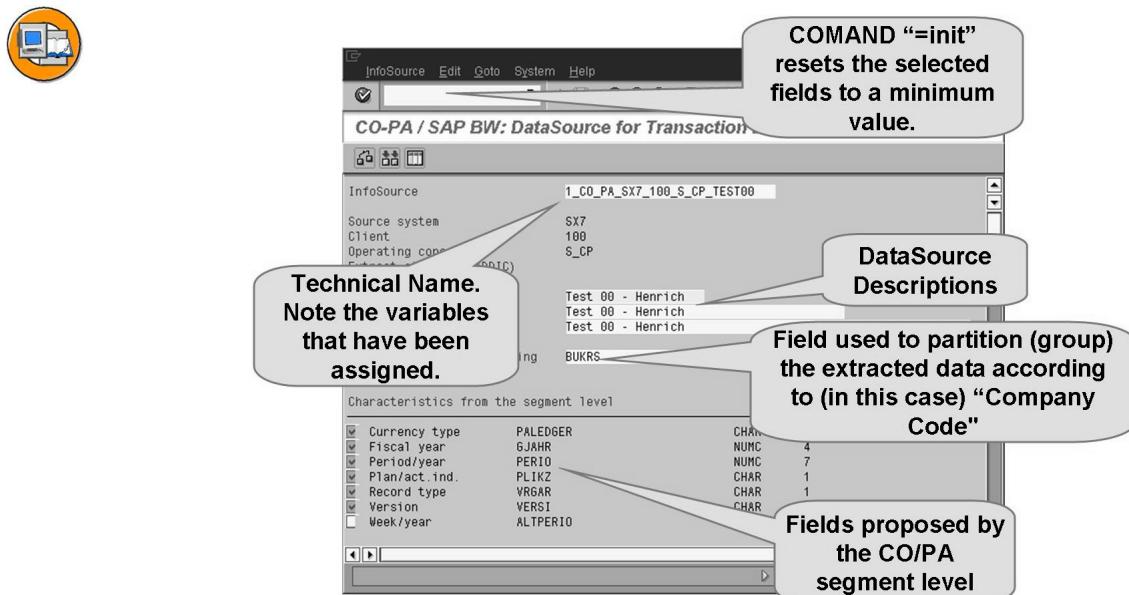


Figure 208: DataSource: Header Information

Some fields of the operating concern are not replicated 1:1 in BI.

- The PALEDDGER field (internal/external values 01/B0, 02/10, 03/B2, 04/12) is divided into currency type (CURTYPE field in the extract structure, InfoObject 0CURTYPE, values B0, 10) and reporting view (VALUTYP, 0VALUATION, values 0, 2).
- The plan/actual indicator PLAID (values 0, 1) is converted into the currency type for the reporting system (VALTP, 0VTYPE, value 010, 020).
- The week ALTPERIO (internal/external values YYYY-WWW/WWW.YYYY) is converted into the calendar week (CALWEEK, 0CALWEEK, internal/external values YYYYWW/WW.YYYY).
- The PERIV fiscal year variant is added to the FISYR fiscal year to describe the PERIO period in more detail.
- With accounting DataSources, the account plan ACCPL is added to cost accounting COACC to describe the cost type CSTTYP.

The name of the DataSource is only maintained in the logon language.

The field name for partitioning can be released.



Characteristics from the segment table		
<input type="checkbox"/> Brand	KMBRND	NUMC
<input type="checkbox"/> Business area	GSSBER	CHAR
<input type="checkbox"/> Business field	KMCATG	NUMC
<input type="checkbox"/> CO area	KOKRS	CHAR
<input checked="" type="checkbox"/> Company code	BUKRS	CHAR
<input type="checkbox"/> Country	KMLAND	CHAR
<input type="checkbox"/> Customer	KNDNR	CHAR
<input checked="" type="checkbox"/> Customer group	KMKDGR	CHAR
<input type="checkbox"/> CustomerHier01	KMH101	CHAR
<input type="checkbox"/> CustomerHier02	KMH102	CHAR
<input type="checkbox"/> CustomerHier03	KMH103	CHAR
<input type="checkbox"/> Distr. channel	VTWEG	CHAR
<input checked="" type="checkbox"/> Division	SPART	CHAR
<input type="checkbox"/> Material group	KMMAKL	CHAR
<input type="checkbox"/> Nielsen ID	KMNIEL	CHAR
<input checked="" type="checkbox"/> Plant	WERKS	CHAR
<input type="checkbox"/> Product	ARTNR	CHAR
<input type="checkbox"/> Profit center	PRCTR	CHAR
<input type="checkbox"/> Sales employee	KMVINR	NUMC
<input type="checkbox"/> Sales group	KMVKGR	CHAR
<input type="checkbox"/> Sales office	KMVKBU	CHAR
<input checked="" type="checkbox"/> Sales org.	VKORG	CHAR
<input type="checkbox"/> Strat. Business Unit	KMST6E	NUMC

Figure 209: DataSource: Characteristics - Segment Table

The characteristics from the segment table are maintained in transaction KEQ3 (characteristics of the profitability segments) and that form the profitability segments.

By default, all the characteristics are selected, since it is based on the scenario “CO-PA Reporting in BI”, that is, “KE30 via BEx”.

Exceptional cases with characteristics:

- Customer hierarchy levels have KNA1 as master data.
- There is a special procedure for compound characteristics. You must create InfoObjects, master data, and text extractors.
- Company code (CCODE) is not a mandatory field, but you must select it if the DataSource is an account-based DataSource.
- You must also specify a controlling area for accounting-based DataSources.
- The WBS element is transferred to BI in its external format (24 characters), since there are no 8-character WBS elements in BI.



CO-PA / SAP BW: DataSource for Transaction Data																																																														
<input checked="" type="checkbox"/> Unit Sales qty KWSVME_ME <small>(Fields for units of measure are selected automatically if the relevant value field has been selected.)</small>																																																														
Characteristics from the line item																																																														
<input checked="" type="checkbox"/> Record Currency REC_WAERS <input type="checkbox"/> Billing type FKART <input type="checkbox"/> Canceled doc. STO_BELNR <input type="checkbox"/> Canceled item STO_POSNR <input type="checkbox"/> Cost element KSTAR <input type="checkbox"/> Cost object KSTRG <input type="checkbox"/> Document number BELNR <input type="checkbox"/> Goods issue WADAT <input type="checkbox"/> Invoice date FADAT <input type="checkbox"/> Item number POSNR <input type="checkbox"/> Order RKAUFRNR <input type="checkbox"/> Partner pr.ctr PPRCTR <input type="checkbox"/> Posting date BUDAT <input type="checkbox"/> Ref.doc.number RBELN <input type="checkbox"/> Reference item RPSON <input type="checkbox"/> Sales ord. item KDPDOS <input type="checkbox"/> Sales order KAUFN <input type="checkbox"/> Sender BProc PRZNR <input type="checkbox"/> Sender cost ctr SKOST <input type="checkbox"/> WBS element PSPNR																																																														
Fields proposed by the line item tables in CO/PA																																																														
<table border="1"> <tr><td>Value fields</td><td>REC_WAERS</td><td>CHAR 10</td></tr> <tr><td></td><td>FKART</td><td>CHAR 6</td></tr> <tr><td></td><td>STO_BELNR</td><td>CHAR 10</td></tr> <tr><td></td><td>STO_POSNR</td><td>CHAR 12</td></tr> <tr><td></td><td>KSTAR</td><td>CHAR 10</td></tr> <tr><td></td><td>KSTRG</td><td>CHAR 12</td></tr> <tr><td></td><td>BELNR</td><td>DATS 8</td></tr> <tr><td></td><td>WADAT</td><td>DATS 8</td></tr> <tr><td></td><td>FADAT</td><td>CHAR 6</td></tr> <tr><td></td><td>POSNR</td><td>CHAR 6</td></tr> <tr><td></td><td>RKAUFRNR</td><td>CHAR 12</td></tr> <tr><td></td><td>PPRCTR</td><td>CHAR 10</td></tr> <tr><td></td><td>BUDAT</td><td>DATS 8</td></tr> <tr><td></td><td>RBELN</td><td>CHAR 10</td></tr> <tr><td></td><td>RPSON</td><td>CHAR 6</td></tr> <tr><td></td><td>KDPDOS</td><td>NUMC 6</td></tr> <tr><td></td><td>KAUFN</td><td>CHAR 10</td></tr> <tr><td></td><td>PRZNR</td><td>CHAR 12</td></tr> <tr><td></td><td>SKOST</td><td>CHAR 10</td></tr> <tr><td></td><td>PSPNR</td><td>NUMC 8</td></tr> </table>			Value fields	REC_WAERS	CHAR 10		FKART	CHAR 6		STO_BELNR	CHAR 10		STO_POSNR	CHAR 12		KSTAR	CHAR 10		KSTRG	CHAR 12		BELNR	DATS 8		WADAT	DATS 8		FADAT	CHAR 6		POSNR	CHAR 6		RKAUFRNR	CHAR 12		PPRCTR	CHAR 10		BUDAT	DATS 8		RBELN	CHAR 10		RPSON	CHAR 6		KDPDOS	NUMC 6		KAUFN	CHAR 10		PRZNR	CHAR 12		SKOST	CHAR 10		PSPNR	NUMC 8
Value fields	REC_WAERS	CHAR 10																																																												
	FKART	CHAR 6																																																												
	STO_BELNR	CHAR 10																																																												
	STO_POSNR	CHAR 12																																																												
	KSTAR	CHAR 10																																																												
	KSTRG	CHAR 12																																																												
	BELNR	DATS 8																																																												
	WADAT	DATS 8																																																												
	FADAT	CHAR 6																																																												
	POSNR	CHAR 6																																																												
	RKAUFRNR	CHAR 12																																																												
	PPRCTR	CHAR 10																																																												
	BUDAT	DATS 8																																																												
	RBELN	CHAR 10																																																												
	RPSON	CHAR 6																																																												
	KDPDOS	NUMC 6																																																												
	KAUFN	CHAR 10																																																												
	PRZNR	CHAR 12																																																												
	SKOST	CHAR 10																																																												
	PSPNR	NUMC 8																																																												

Figure 210: DataSource: Characteristics - Line Item Table

The units for the corresponding quantity fields that have been selected are assigned automatically.

Characteristics from single line items

- In this area, all the characteristics are displayed that were not selected for the operating concern in transaction KEQ3, for example, fields that are saved only in single line items.
- Only those fields for single line items that have standardized data in the database tables are available. Some fields are either redundant (for example, CLIENT), or they are intended for CO-PA internal use only (for example, PAOBJNR, AWSYS), or they do not contain any relevant values. Note that some of the fields shown are not relevant for customer installation (for example, STO_BELNR, RBELN).
- Remember that all updates (even full updates) get their data from the single line items if you choose a field from this area. Delta updates always get their data from the line items. Whether or not you select a line item characteristic is of no consequence here.
- Selecting the document number (DOCNR) leads to non-aggregated data being loaded into BI. When you save document numbers, remember that this will reduce the system performance during the extraction process in the source system and during the update process within BI, as well during the runtime of the queries.



CO-PA / SAP BW: DataSource for Transaction Data				
Value fields				
<input type="checkbox"/>	Administration	KWVERW	CURR	15 REC_WAERS
<input type="checkbox"/>	Advertising	KWMKAD	CURR	15 REC_WAERS
<input type="checkbox"/>	Anticipd ship. costs	KWKLFK	CURR	15 REC_WAERS
<input type="checkbox"/>	Bonuses	KWBONI	CURR	15 REC_WAERS
<input type="checkbox"/>	Cash discount	KWSKTO	CURR	15 REC_WAERS
<input type="checkbox"/>	Customer discount	KWKORB	CURR	15 REC_WAERS
<input type="checkbox"/>	Direct mat. costs	KWMAEK	CURR	15 REC_WAERS
<input type="checkbox"/>	Fixed manuf. costs	KWFKFX	CURR	15 REC_WAERS
<input checked="" type="checkbox"/>	Gross sales	KWBRIH	CURR	15 REC_WAERS
<input type="checkbox"/>	Logistics	KWBISI	CURR	15 REC_WAERS
<input type="checkbox"/>	Other variances	KWAESO	CURR	15 REC_WAERS
<input type="checkbox"/>	Marketing division	KWHDP	CURR	15 REC_WAERS
<input type="checkbox"/>	Mat. overhead costs	KWMASK	CURR	15 REC_WAERS
<input type="checkbox"/>	Material discount	KUMARB	CURR	15 REC_WAERS
<input type="checkbox"/>	Other variances	KWAESS	CURR	15 REC_WAERS
<input type="checkbox"/>	Price variance	KWAOPR	CURR	15 REC_WAERS
<input type="checkbox"/>	Prod. Costs Variance	KWABLX	CURR	15 REC_WAERS
<input type="checkbox"/>	Promotion	KWMKPR	CURR	15 REC_WAERS
<input type="checkbox"/>	Quantity discount	KWMGRB	CURR	15 REC_WAERS
<input type="checkbox"/>	Quantity variance	KWABMG	CURR	15 REC_WAERS
<input type="checkbox"/>	Research & Developmt	KWFQEN	CURR	15 REC_WAERS
<input type="checkbox"/>	Sales commission	KWVKPV	CURR	15 REC_WAERS
<input type="checkbox"/>	Sales field service	KWTAD	CURR	15 REC_WAERS
<input type="checkbox"/>	Sales office	KWTID	CURR	15 REC_WAERS
<input checked="" type="checkbox"/>	Sales quantity	KWVRE	DUAN	15 REC_WAERS
<input type="checkbox"/>	Sales special costs	KWSEK	CURR	15 REC_WAERS
<input type="checkbox"/>	Ship/trpt variances	KWAFK	CURR	15 REC_WAERS
<input type="checkbox"/>	Spec. offer discount	KWAKRB	CURR	15 REC_WAERS
<input type="checkbox"/>	Structure variance	KWABST	CURR	15 REC_WAERS
<input type="checkbox"/>	Variable manuf. costs	KWFKYA	CURR	15 REC_WAERS

Figure 211: DataSource: Value Fields

Finally, select the value fields (run as key figures in BI later on) that are needed for the analysis.

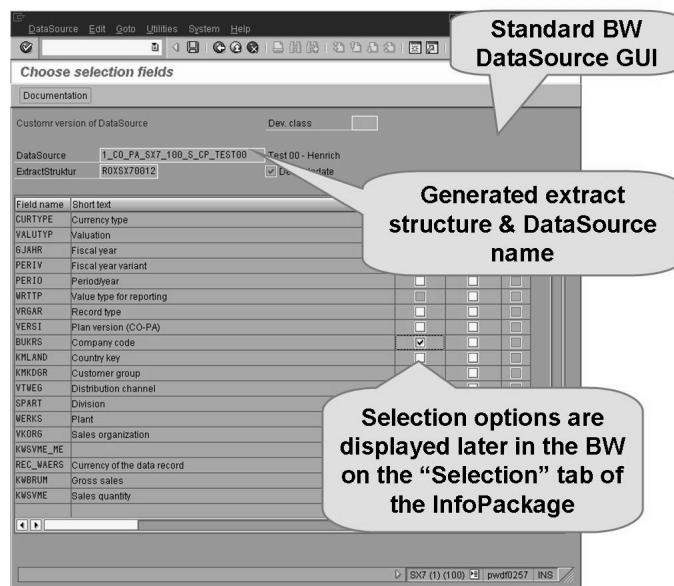


Figure 212: DataSource: Selection Fields

If the checkbox in the *Selection* column is active, meaning if it is displayed in white, you are able to select it.

- Some fields cannot be selected, however:
Fields to be dealt with in a special way (currency, PALEDDER, PLIKZ, PERIV, KTOPL, and so on), units, currency, value fields, and calculated key figures.
- Select fields that partition your data (in the operating concern) only if (a) there is a large volume of data for the first data transfer, or (b) there are special business reasons for selecting the fields.
- The selection in BI can be a fixed value or an interval. Variable changes to the selection values during background processing (ABAP, for example.)
- Never specify very large PERIO intervals (PERIO=001.1999..999.2999). The system will crash because the memory is overloaded. Use FYEAR intervals instead (FYEAR=1999..2999).

You can ignore the *Hide Field* column, since you have already selected the fields that you want to replicate.

After making your selections, choose *Generate*. This ends the activities in the SAP system.

Choose the detail view in the IMG to display the DataSources.

The next steps are made in the BI System. First, replicate the DataSource that you have recently defined in BI and generate an InfoSource and an InfoCube based on this DataSource or InfoSource.

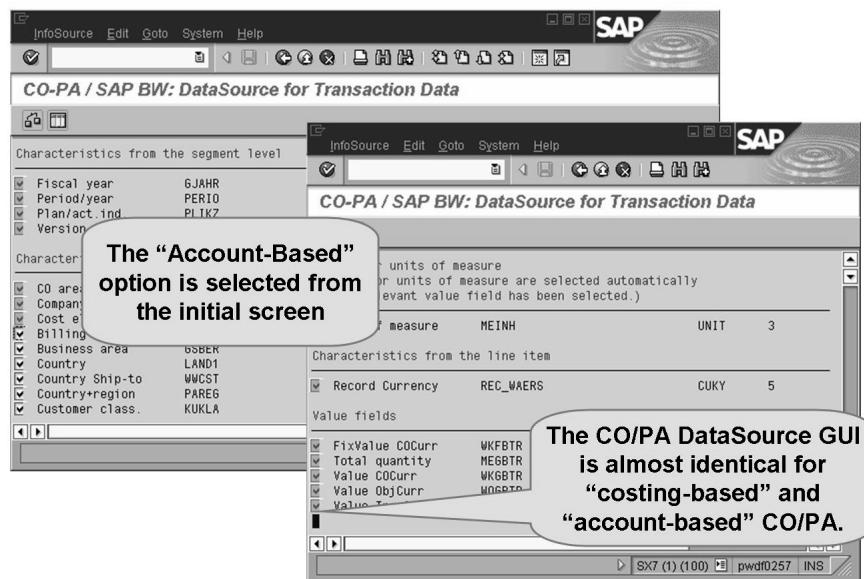


Figure 213: DataSource Definition: Account-Based CO-PA

You define an account-based CO-PA DataSource in almost the same way as you would define a costing-based CO-PA DataSource.

When you generate a CO-PA DataSource, select the “account-based” option from the first screen.

- KOKRS, BUKRS, KSTAR are compulsory.
- There are no characteristics available from the line items, because account-based DataSources do not contain characteristics.
- There is only one amount (in three currencies) and a quantity (together with the quantity unit).
- There are no value fields or calculated key figures available.
- KOKRS and PERIO must be selected as selection fields.

There are certain restrictions for the data upload. These are described in more detail in the section on additional update topics.



Account-based CO-PA:

Method	Line Items	Segment Level	Summarization Level
Delta	V	-	-
■ Init. load	-	-	V
■ Delta	V	-	-
Full	-	-	V

In account-based CO-PA, data is read directly from CO and there are no CE1-3 tables, which explains the read mode restrictions.

Costing-based CO-PA:

Method	Line Items	Segment Level	Summarization Level
Delta			
■ Init. load	V	V	V
■ Delta	V	-	-
Full	V	V	V

**Figure 214: Data Sources for Extracting Data from CO-PA to BI:
Account-Based CO-PA**

The extractor always tries to select the most appropriate data source, meaning the DataSource with the smallest quantity of data. The following restrictions apply:

- Up to and including Release PI2001.1, only the “full” update mode is supported from the summarization levels for extractions from the account-based profitability analysis. This means that you can only load individual periods for a controlling area. From Release PI2001.2 the delta process can also be used. As before, the initialization must start from a summarization level. Subsequent delta updates read line items. In the InfoPackage, the controlling area remains a mandatory field. This means it is no longer necessary to select from individual periods. The period is, however, still a mandatory field in the selection. If you do not want this, follow the instructions in SAP Note 546238. To read data from a summarization level, all the characteristics to be extracted with the DataSource must also be contained in this level (entry * in the maintenance transaction KEDV). The summarization level also needs to be 'ACTIVE' (this also applies to the search function in the maintenance transaction for CO-PA DataSources, KEB0)
- With DataSources from the costing-based profitability analysis, data can only be read from a summarization level if no other characteristics from the line item other than the “record currency” (REC_CURR) field (always selected) have been selected. As with summarization levels, an extraction from the segment level, meaning from the combination of the tables CE3XXXX / CE4XXXX ('XXXX' being the name of the operating concern) is run for full updates only if no line item characteristics have been selected. For the delta method and subsequent delta updates, the data has to be read up to a specified point in time. When the delta process is initialized, there are two possible sources: Summarization levels lead the time of the last update / reconstruction of the data. If no line item characteristics have been selected, and a suitable active summarization level (see above) is available, the DataSource “inherits” the time information from the summarization level. If the system is unable to read data from a summarization level, it reads line items instead. Since data can continue to be posted during the extraction, the segment level is not a suitable data source. During the extraction further updates can follow for profitability segments that have been posted already. These values would have to be worked out again by reading the line items, and this would extend the duration of the extraction considerably. With delta updates, data is always read from line items. When you read data from line items, make sure that the indexes from SAP Note 210219 have been created for both line item tables CE1XXXX (actual data) and CE2XXXX (planned data). Otherwise, selections will take a long time to complete. As of release 4.5, suitable indexes for archiving are delivered in the dictionary. These indexes are shipped with the SAP standard system, but you still have to create them on the database.



Changes to assignments in OLTP invalidate the delta status of the DataSource and make it necessary to re-initialize the delta.

If a large volume of data makes initialization impossible or if the consistency of the data between OLTP and BI is not considered important, the FLAG_DIRTY field can be set for the DataSource in the TKEBWL table.

One possible application would be if data from the OLTP always had to be read "as posted".

Figure 215: Realignments in CO-PA and the Effect on BI Data Extraction

Reassignment in the SAP system with reverse method: In Releases earlier than 4.* there is an option that allows you to make reassessments using a reverse method. All changes to characteristics on the line item level are carried over so that you do not need to re-initialize the delta. The disadvantage of this method is that it lengthens the runtime and increases the volume of data. It is therefore only useful for small reassessments. As of release 4.*, a function to undo reassessments was introduced, making the reversal method defunct. This is why the delta has to be re-initialized from the BI following a realignment in the OLTP. If there is a large volume of data that makes an initialization impossible, or if the consistency of the data between OLTP and BI is not considered an important issue, the RD145 error message can be suppressed for the affected DataSource. For this the FLAG_DIRTY field in the TKEBIL table has to be set for the affected DataSource. You are not permitted to maintain the TKEBIL table, so that the easiest way to make the corresponding change is by using a short program. One possible instance where this could be used would be if data from the OLTP always had to be read "as posted". In the detailed display for CO-PA DataSources in transaction KEB2, **With Realignment** is displayed in the header information. This entry is for information purposes only, to remind you that you have to re-initialize the delta process if you make a realignment.

In this step, the definition of the DataSource that you just created is loaded into a BI System.

We recommend that you start the replication process at the most detailed node to obtain the optimum response time. You should only replicate the entire SAP OLTP system or a high-level node if you are creating nodes for new application components, because the process is very time-consuming.

Always use the lowest level of the tree that contains the DataSource that you want to replicate.

You can use the *Modeling - DataSource* tab in the application component you are working in to create a Data Transfer Process.

Note the following:

- If they do not exist already, InfoObjects are created for CO-PA fields (WW..., VV...) that have names consisting of 5 characters. These InfoObjects are named using the naming conventions OG_Cxxxxx (characteristics), OG_Axxxxx (amounts), OG_Qxxxxx (quantities), and OG_Uxxxxx (quantity units). Here, xxxx stands for the name of the relevant CO-PA field. These InfoObjects are assigned automatically to the corresponding fields in the DataSource. Example: VV007 / VV007_ME « 0G_QVV007 / 0G_UVV007.
- Fields from the template operating concern (S_CP, S_GO, S_AL) from 0MB1 (banking) and from the staged key figures all have access to relevant Business Content InfoObjects that are automatically assigned to them. Note that you must install Business Content before you assign the DataSources. Example: KMKDGR « 0CUST_GROUP.
- Fields that are mapped uniquely in other DataSource/InfoSource mappings, inherit this same mapping. Example: CCODE « 0COMP_CODE.
- You can use the standard procedure used in BI to design or generate missing InfoObjects. .

CO-PA DataSources support the delta process.

After the system has retrieved the data from the line item or from the summarization levels, it aggregates it, calculates the key figures, and deletes the records with zero values. This is why the number of records that have been read and the number of records that are transferred into BI (both are displayed in TA KEB2) sometimes differ considerably.

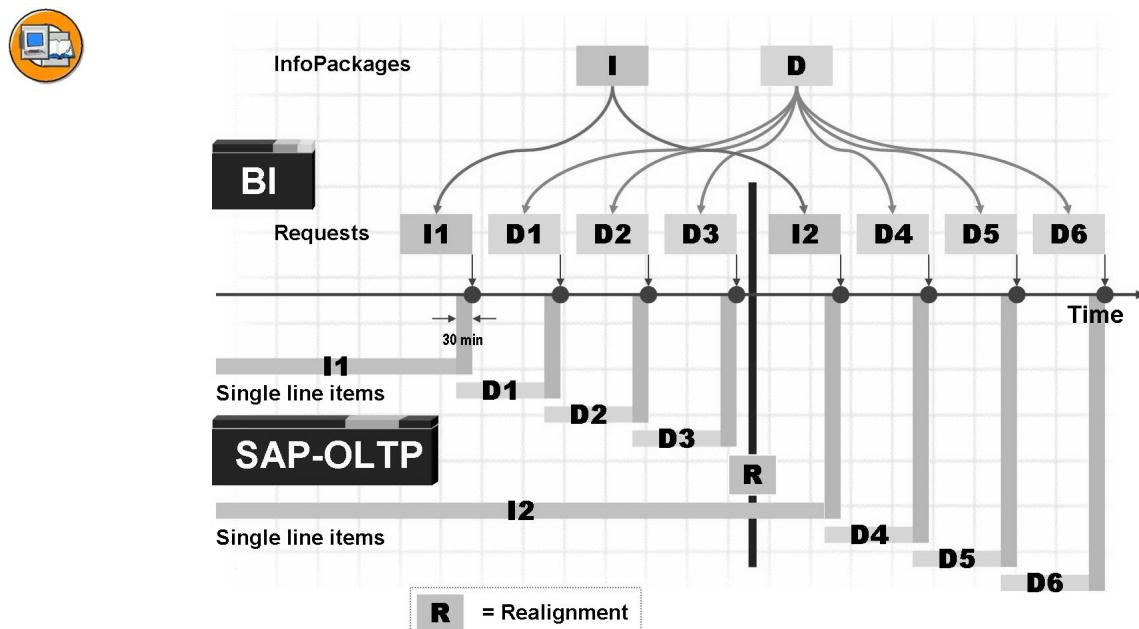


Figure 216: Delta process

The delta process is valid for both costing-based and account-based CO-PA DataSources. As of release PI2004.1 (release 4.0 and higher) there is an additional new logic (generic delta) for the delta process. Old DataSources can be converted to the new logic. New DataSources automatically use the new logic. With the new logic, time stamp management is no longer in profitability analysis, but in the Service API.

“I” stands for initialization request; “D” stands for delta request.

A delta initialization or delta request from line items replicates all the new line items that have been posted previously at the time ($t = 30$ minutes). This guarantees the integrity of the data. The reason is that time stamps can be slightly different across different application servers.

This retention period of 30 minutes is often referred to as a safety delta. It is displayed as yellow bars in the schema. The system only extracts data that is at least 30 minutes old.

Since a delta update can be scheduled every night, after the day's postings, the retention period does not disrupt business during the day.

This safety delta is also used when CO-PA summarization levels are generated. As mentioned previously, initializations from summarization levels are much faster than from line items or CE3 and 4. Make sure that the summarization level is up-to-date. Otherwise the first delta run following the initialization will be much longer.

The screen below shows that the initialization process has to be repeated when you reconstruct the CO-PA schema following a change to the business rules that determine or assign costs and revenue. This does not apply if all the changes that affect BI have been modeled as navigation attributes.

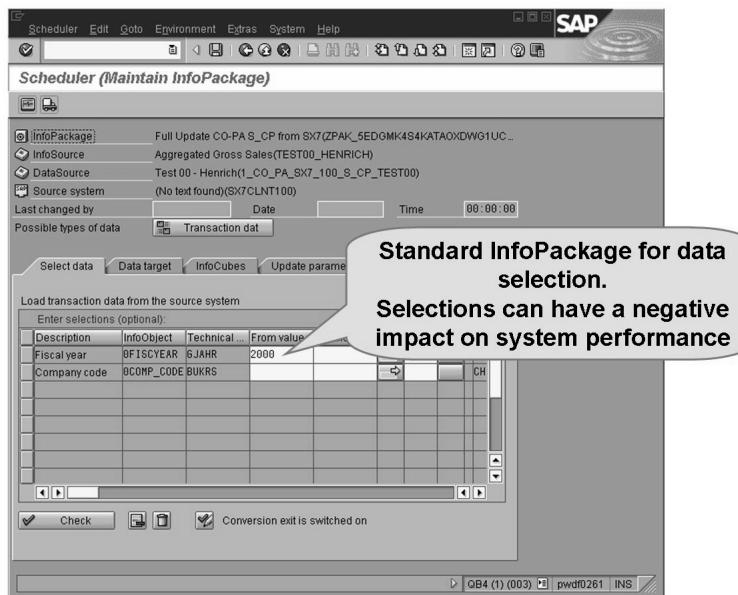


Figure 217: BI: Selection Restriction

Full updates can be carried out with various selections; however, in a cycle in which the delta initializations and delta updates alternate, the selections must be the same for all requests.

Note that the index structure in the CE1xxxx-, CE2xxxx- and CE4xxxx tables can lead to problems with the system performance.

- The extra index CLNT+PAOBJNR for CE1xxxx and CE2xxxx supports all types of requests provided that no selections are used.
- Situations may arise where delta initializations and delta requests cannot be dealt with effectively with a single index structure for the CO-PA tables.
- We recommend that you test this using live data.



Lesson Summary

You should now be able to:

- Design and define DataSources in CO-PA
- Build an InfoCube and maintain transformation rules
- Manage CO-PA extraction
- Use CO-PA tools for BI

Lesson: Data Acquisition from Special Ledger (Optional)

Lesson Overview

This lesson explains how data is acquired from financial areas such as Special Ledger (FI-SL).



Lesson Objectives

After completing this lesson, you will be able to:

- Position FI-SL in the SAP system
- Describe FI-SL data structures
- Generate FI-SL DataSources
- Explain the particular methods used to extract data from FI-SL
- Create generic DataSources for set hierarchies

Business Example

Your company uses Special Ledger (FI-SL) and wants to use this data in BI.

You are the person responsible for extracting data from the SAP OLTP system, and importing it into BI. You therefore have to create the corresponding components in the SAP system and in BI.

Special Ledger in the SAP Landscape

In this section you will learn about the flow of data between other areas in SAP and Special Ledger.

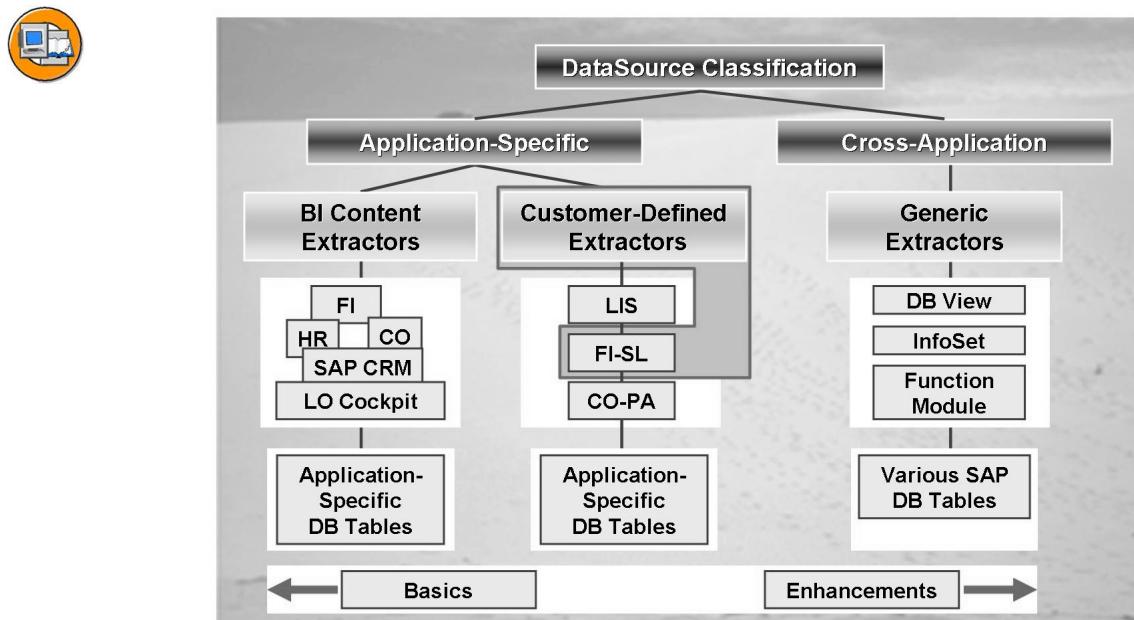


Figure 218: Extracting Data from FI-SL: Overview Diagram

FI-SL is a system in which data (planned and actual) from different OLTP applications (for example, FI-GL, EC_PCA, CO-CCA) can be combined at different levels. FI-SL enhances the functions and usability of these applications in R/3 without restricting their operative functions. FI-SL includes planning functions and reporting tools, but is restricted in that cross-application reporting is not as differentiated as BI reporting and the OLTP system is optimized for transaction processing and high reporting workload would have a negative impact on the overall performance of the system. To overcome these restrictions, FI-SL reporting in BI is recommended.

To understand the processes of extracting and updating data from the SAP system FI-SL into BI, we will first take a closer look at the functions in the FI Special Ledger system.

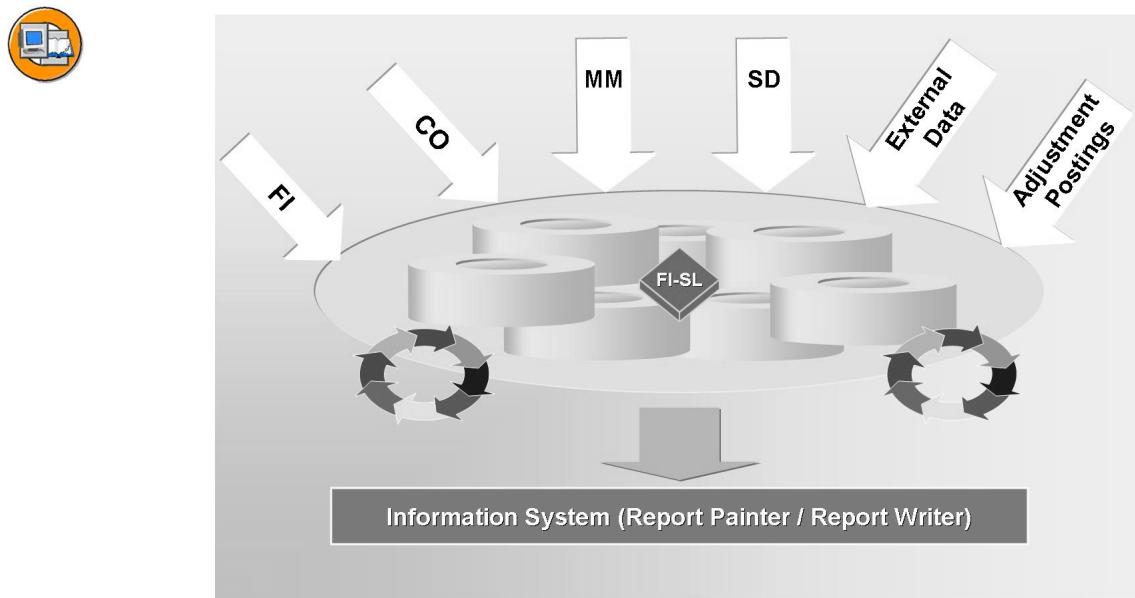


Figure 219: FI-SL: Interaction with Other Applications

In the Special Ledger application, you can define your own ledgers for reporting purposes. You can manage these ledgers with any account assignment objects you like. Account assignment objects can be SAP dimensions from the various applications (such as account, cost center, business area, profit center) and dimensions of your own (such as region).

Special ledgers enable you to report across several levels on the values from the various applications. The modules available in the special ledgers give you several options for manipulating the data that has been transferred from other SAP applications and external systems into the special ledgers, for example, collecting information, combining information, forming totals, modifying totals, and moving or distributing actual and planned values.

The operative functions of other SAP applications are not affected when you use the special ledgers.

The special ledger system (FI-SL) is a receiver system, in which data from other SAP applications can be stored. It is not a sender system for other SAP applications. In FI-SL, you can use flexible data structures (with additional fields) that correspond to individual business requirements. This allows you to combine and summarize information for any number of combinations at the required level of detail. Up to 3 currencies and 2 quantities can be managed in parallel in FI-SL. The assignment of transaction to particular combinations of company code/ledger or compay/ledger determines which ledgers are updated using the data from the applications.

Data Flow and Structures in Special Ledger

The basic concepts behind the SAP FI-SL system are as follows.

The addition of an extra field to a financial accounting document is an example of a flexible data structure. This field can be filled either in the application when the document is posted, or through a data update in FI-SL.

A ledger is updated either at company code level or at company level. Company codes are defined in the FI system and are used more often than companies in the FI-SL system.

- You can configure your system so that a local ledger is updated directly through a company code (“Ledger/Company Code” assignment).
- A company normally comprises several company codes. You can assign a company code to a company so that a local ledger and a global ledger are updated through the company code (“Ledger/Company” assignment).

A transaction describes a business transaction in the SAP system. Examples of business transactions include **goods receipt** in Materials Management (MM), **document posting** in Financial Accounting (FI), and so on.

Adjustment postings (direct data entry) can be made in the FI-SL system. In FI-SL, you are able to use alternative charts of accounts (operative, group-specific, and country-specific charts of accounts). Various fiscal year variants enable you to create weekly or monthly reports. Validations or substitutions allow you to check or modify the validity of the data when it enters the FI-SL system.

Adjustment postings by direct data entry allow you to do the following:

- Post documents exclusively to the FI-SL system, post different versions of documents.
- Enter additional currency amounts manually.
- Post statistical entries that only affect quantities.
- Post documents with a balance not equal to zero.

The fiscal year variant determines the number of periods in a fiscal year. This does not have to be the same number of periods as there are in the calendar year. Banks and other financial institutions might want a daily ledger, whereas the general ledger (FI-GL) produces monthly reports.

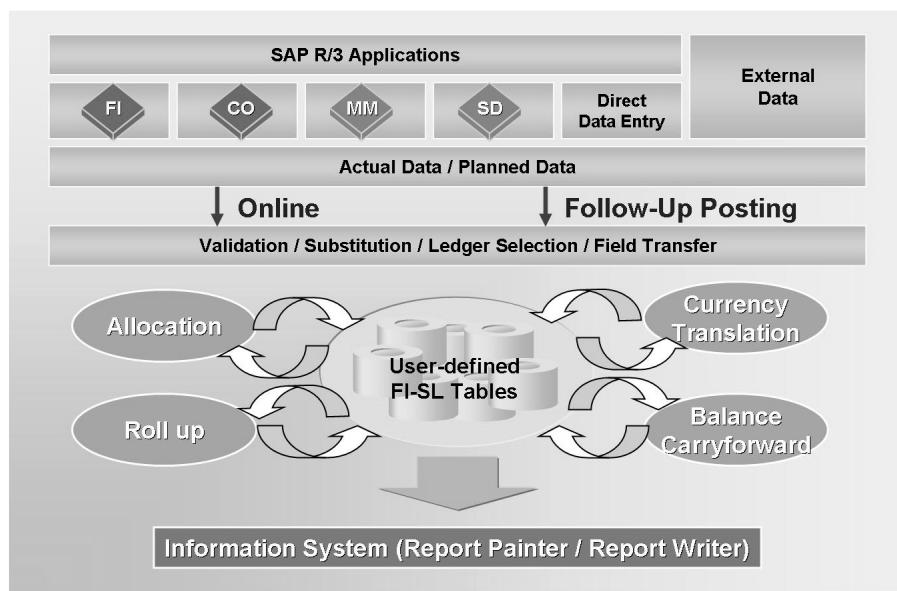


Figure 220: Data Flow in FI-SL

In addition to data from FI, CO, MM, and SD, external data and data that is entered directly can also be posted into FI-SL.

The update takes place either online or as a post-run update. In the case of the latter, a predefined number of data records are transferred to the FI-SL database tables at a specific point in time, regardless of the time the original documents were created.

You can use various functions to update data in the FI Special Ledger:

- *Validation* uses freely definable rules to check the data to be updated
- *Substitution* replaces the data with modified data according to a set of rules before the update is run
- *Ledger Selection* allows data from a transaction to be updated in selected ledgers

The fields that are transferred determine which characteristics and dimensions are used in Special Ledger. You use these dimensions when you design your special ledger.

Operations available for FI-SL data:

- The *Currency Translation* function translates amounts that have been posted to ledgers in the FI-SL system.
- At the end of the fiscal year, you use the *Balance Carryforward* function to transfer actual values and plan values from the previous fiscal year to the new fiscal year.
- *Allocation* (assessment/distribution) is the transfer of an amount that has already been posted from a sender-object to one or more receiver-objects (for example, distribution of overhead costs across several cost centers).
- You can create *rollup* ledgers containing cumulated and summarized data from one or more of the other ledgers to speed up report processing times (comparable with aggregates in BI).

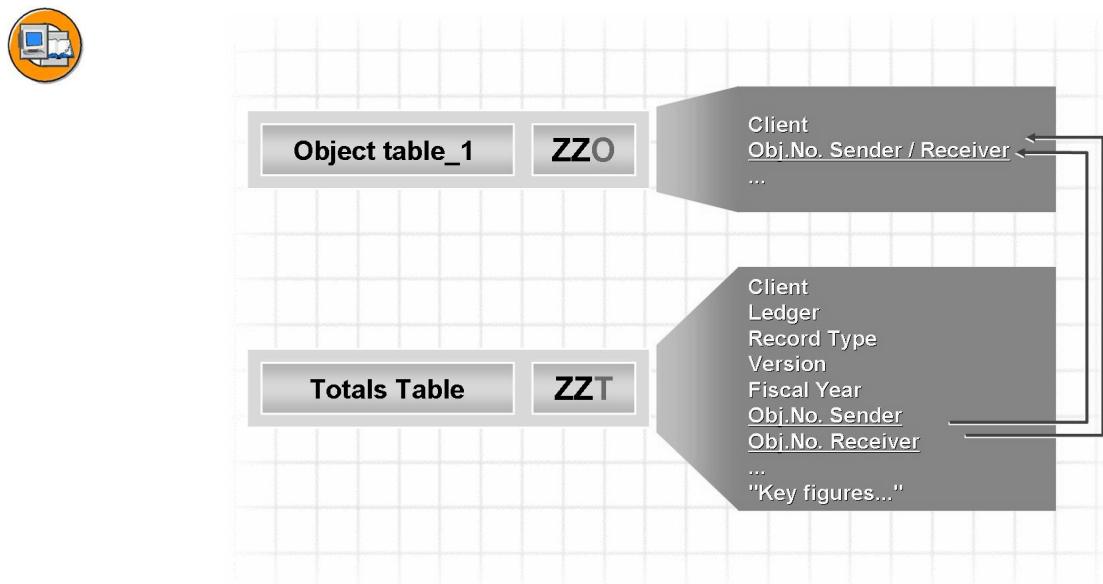


Figure 221: Data Structures in FI-SL: Table Group ZZ

When you create an FI-SL table group (after you have set the organizational unit to company code or company) up to 5 tables with fixed naming conventions are created: Summary table (...T), actual line items tables (...A), plan line item tables (...P), object table_1 (object/partner) (...O), and the optional object table_2 (movement attribute) (...C).

In the object table, an object number is assigned to a combination of characteristics (similar to dimension tables in BI). The key figure values are stored in the summary table along with the resulting object numbers (similar to the fact table in BI).

Ledgers that refer to this kind of table group are designed (with the summary table in the center) to enable you to report on the contents of these tables.

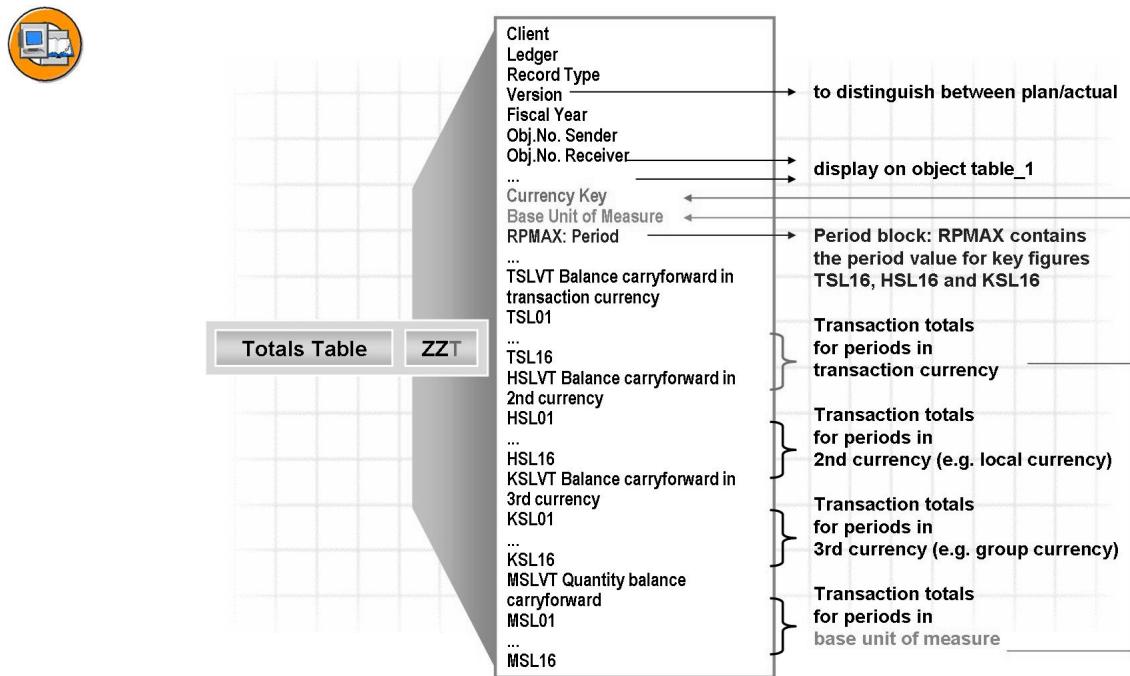


Figure 222: Totals Table ZZT in FI-SL

The period block is a particular feature of data modeling in the summary table: There is a key field RPMAX (period) that specifies the meaning of the period for the key figures TSL16, HSL16, and KSL16. The meaning of the periods for the key figures *SL01 to *SL15 is derived from this.

If the value “016” appears in RPMAX, the values of the key figures TSL16, HSL16 and KSL16 refer to period “016” for the fiscal year variant that you set in Customizing.

If the value “064” appears in RPMAX, the values for the key figures TSL16, HSL16, and KSL16 refer to period “064”, the values for the key figures TSL15, HSL15, and KSL15 to period “063”, and so on, for the fiscal year variant that you set in Customizing.

In this way you can map the fiscal year variants with a period number > 16; the key figure values are distributed among various data records with different RPMAX values.

The handling of currencies and quantities is explained here too:

- The key figures TSL01 to TSL16 are always assigned the transaction currency that is specified in the field RTCUR (currency key).
- The key figures HSL01 to HSL16 are assigned the second currency. This is specified in Customizing and could be the local currency, for example.
- The key figures KSL01 to KSL16 are assigned the third currency. This is also specified in Customizing and could be the company code currency, for example.
- The key figures MSL01 to MSL16 are assigned the quantity value.

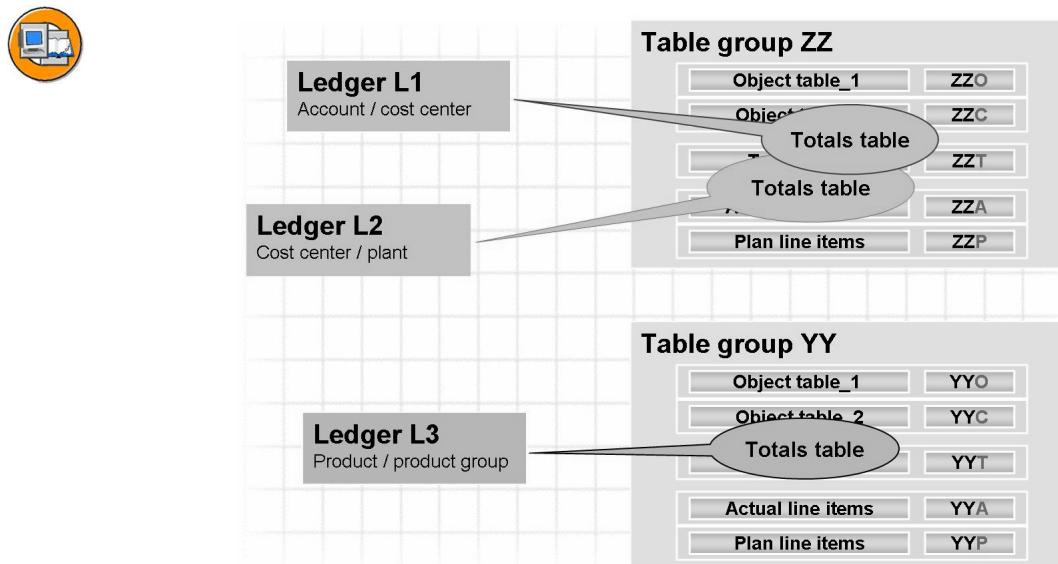


Figure 223: Ledgers in FI-SL

A ledger is a closed reporting area that always refers to one particular summary table. You can consider a ledger to be the logical part of a summary table.

You can define several ledgers for a summary table.

A ledger is created for a summary table in Customizing. The ledger contains metadata, such as the set of rules by which it is updated.

A maximum of 4 different sender-structures (MOVE or programmable Exit) can update a ledger. Manual adjustment postings are also possible.

The transaction data content itself (physical storage of the data) always remains in the tables of the table group.

DataSources in Special Ledger

The FI-SL transaction data is stored in a group of tables. A ledger can refer to a selection of fields in a table group. The object numbers in the summary table are not “meaningful” in that the period block means that the key figure (TBLD1, for example) can refer to two different totals records in two different periods. The currency information is only partially stored in the table group.

Here are some of the difficulties that are overcome by generating FI-SL DataSources.

When you generate an FI-SL DataSource (a ledger based on a summary table) the system proposes a combination of characteristics and key figures from the summary table.

The difficulties outlined on this slide are explained in detail on later slides. The important thing here is that the difficulties involved in extracting data from FI-SL are “invisible” to the customer, because SAP provides a program (extractor). This program provides a solution to these problems and extracts the FI-SL data to the extract structure and then into BI.

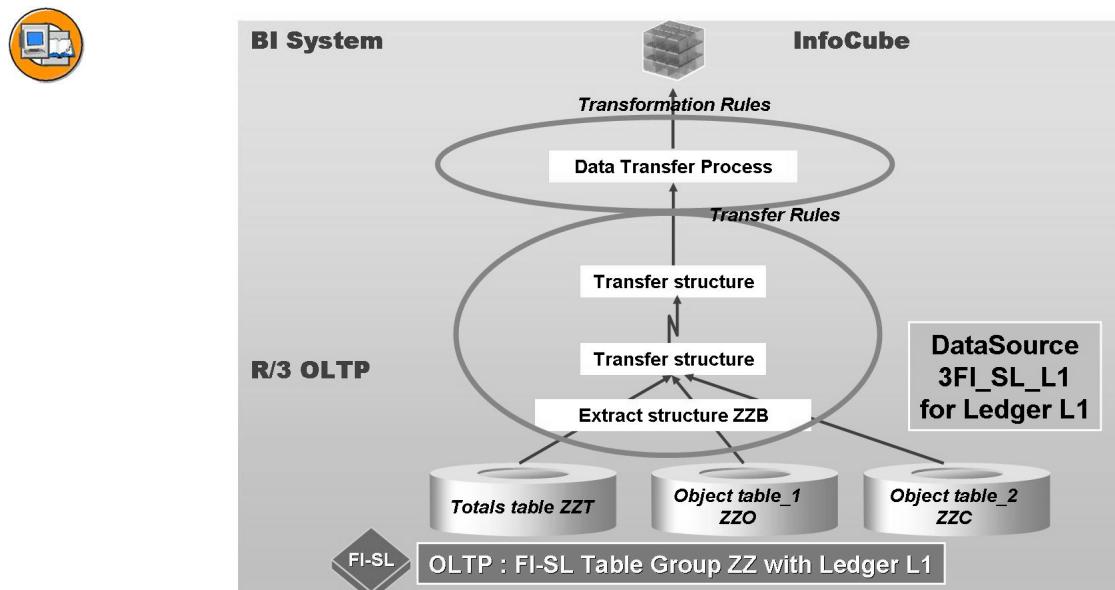


Figure 224: Updating the FI Special Ledger in BI

The BI DataSources prepare the data that is going to be updated in the Business Information Warehouse.

SAP is not able to ship FI-SL DataSources with BI Business Content for non-standard ledgers, because the ledgers and their FI-SL summary tables might include user-defined fields and structures.

All other FI DataSources (FI-GL, FI-AR, FI-AP) are delivered with Business Content.

To provide an FI special ledger DataSource for BI, you have to generate an extraction structure for the FI-SL summary table in question. Finally, you have to define the DataSource for the summary table ledger.

These steps will now be explained in more detail ...

The resulting DataSources transfer the totals records from FI-SL. No journal entry records are transferred into BI.



Hint: Note: In some BI Releases, in the menu paths for FI-SL, the name FI-SL “Transfer Structure” is sometimes used instead of “BI Extraction Structure”. It is the “extraction structure” that is being generated, however.



(in the BW IMG of the OLTP system)

1. Generating an Extract Structure ZZB for a Totals Table ZZT

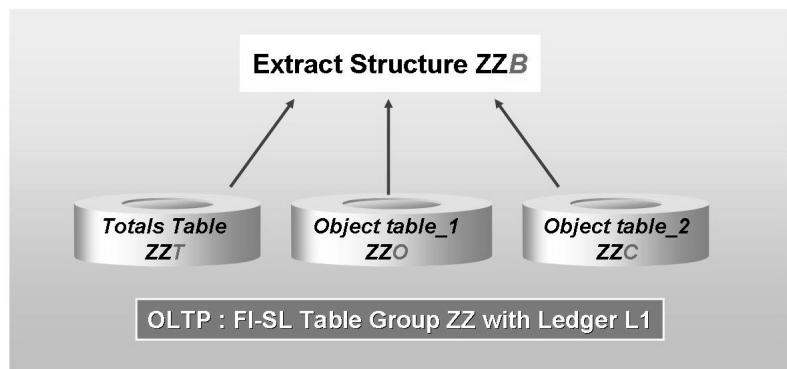


Figure 225: Setting Up an FI-SL DataSource (Step 1)

As mentioned above, the phrase “generate an FI-SL transfer structure” appears in the menu paths in some “Plug-In” releases. It is the “extraction structure” that is being generated, however.

Acquiring Data from Special Ledger

You need to perform the following Customizing activities in the OLTP system to enable you to transfer transaction data in the ledgers you have generated yourself from the source system OLTP into the Business Information Warehouse:

- You use transaction /NSBIW to call the BI IMG for the OLTP system:
 - *Business Information Warehouse → Settings for Application-Specific DataSources*
 - *Financial Accounting: Special Ledger*
- Step 1: Generate transfer structure (= **extraction structure**) for summary table.

In step 1, you generate a transfer structure for your summary table. This structure is used to transfer transaction data from your ledgers into the Business Information Warehouse.

The structure contains fields from the summary table, fields from the object table, and fields derived from Customizing.

The system names the extraction structure according to the following naming convention for summary tables:

- If the name of the summary table ends in T, the T is replaced with a B.
- If the name of the summary table does not end in T, a B is added as a suffix to the name of the summary table.

If this summary table is refreshed, the existing structure is overwritten.



(in the BI IMG of the OLTP system)

2. Defining and Assigning a DataSource 3FI_FL_L1 for a Ledger L1

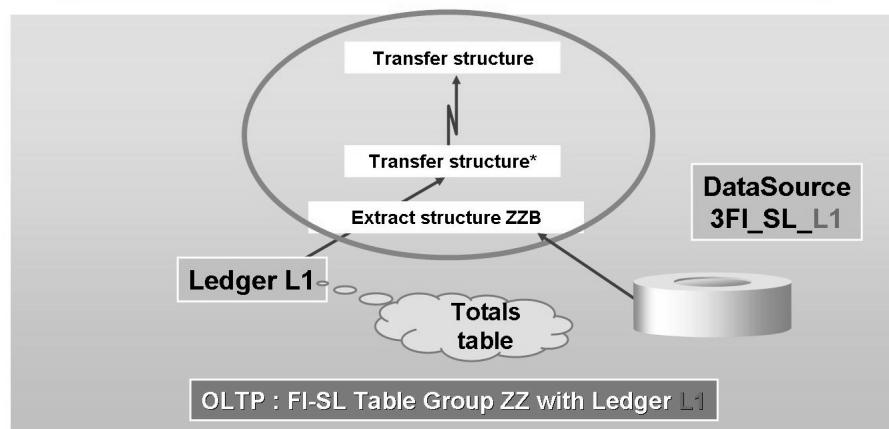


Figure 226: Setting up an FI-SL DataSource (Step 2)

You use transaction /NSBIW to start the BI IMG for the OLTP system:

1. In the Implementation Guide for BI, choose *Business Information Warehouse* → *Settings for Application-Specific DataSources*. Choose *Financial Accounting: Special Ledger*.
2. Create DataSource for ledger. Define a DataSource for a ledger and assign this DataSource to the ledger. In this step, all the non-standard ledgers with summary tables that have already been assigned an extraction structure are displayed first.



Note: The ledger may not display for the following reasons:

- The ledger is a standard ledger (starts with a number, for example, 00=General ledger, 0F=Sales costs.) There are Business Content DataSources for this ledger.
- The summary table for the ledger does not have an extraction structure.

A green traffic light in the *status* column indicates that a DataSource has been assigned to the ledger.

3. Select the ledger to which you want to assign a DataSource. Type in the name of the DataSource that you want to assign to the ledger.



Note: Follow this naming convention: The name must start with the number 3.

The system proposes a name with the prefix 3FI_SL_ and the name of the ledger.



Hint: You are not able to change the name after you have saved the assignment.

Ledgers are client-specific, whereas DataSources are cross-client objects. Within a client, a DataSource can be assigned to only one ledger.

The DataSource delivers transaction data from an FI-SL ledger at the totals record level.



Screenshot of the SAP GUI showing the 'DataSource: Customer Version Edit' window. The window includes a toolbar with icons for DataSource, Edit, Goto, Utilities, System, and Help. The main area displays the extraction structure for DataSource 3FI_SL_LY.

Header Data:

DataSource	3FI_SL_LY
Description	

Extraction:

ExtractStruct.	GLFUNCB
Direct Access	2
Delta Update	<input type="checkbox"/>

Field name Short text Selection Hide field

RVERS	Version	<input type="checkbox"/>	<input type="checkbox"/>
RTCUR	Currency Key	<input type="checkbox"/>	<input type="checkbox"/>
RBUKRS	Company Code	<input type="checkbox"/>	<input type="checkbox"/>
RACCT	Account number	<input type="checkbox"/>	<input type="checkbox"/>

Figure 227: FI-SL DataSource

This overview shows some of the fields in the extraction structure of DataSource 3FI_SL_LX.

A *global ledger* has the company as its organizational unit. The *company* characteristic appears in the transfer structure only with DataSources for global ledgers. With global ledgers, the *company code* characteristic is optional.

Other comments on DataSources:

- *Chart of Accounts* and *Fiscal Year Variant* cannot be used as selection criteria.
- The *Version* field shows the version managed in FI-SL.
- The FI-SL *Record Type* characteristic is mapped to the *Reporting Value Type* InfoObject. The extractor transforms 2 reporting value types from 4 different FI-SL record types: Plan and actual.

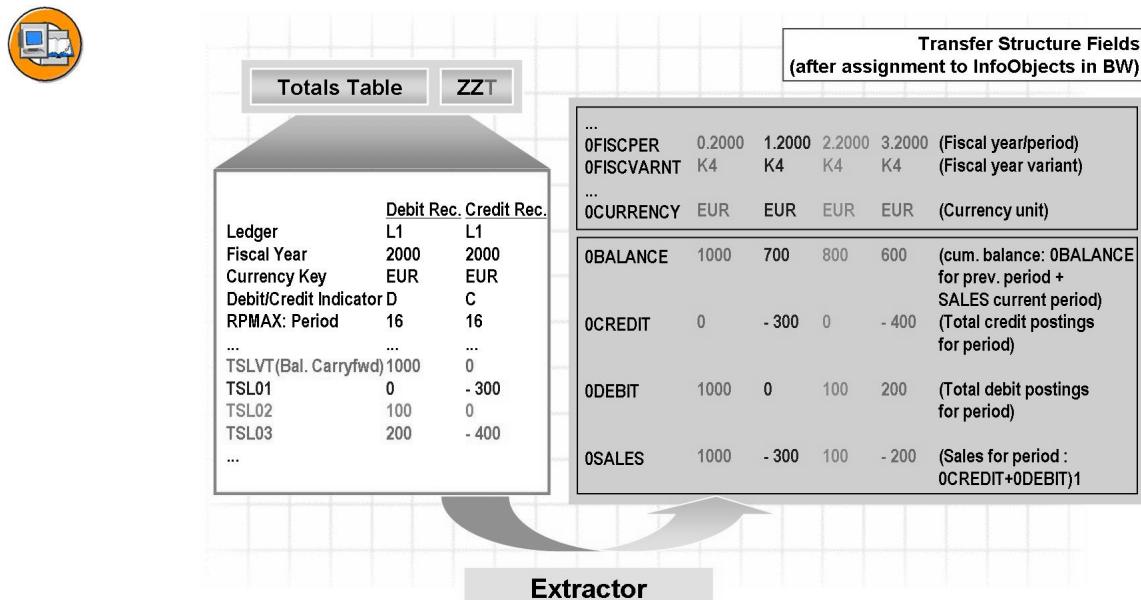


Figure 228: Extractor: Determining Key Figure Values

This example shows how the extractor uses the key figure values from an FI-SL summary table. The values that the extractor determines in the format of the transfer structure result from the following:

- The time characteristic values are determined in the transfer structure from each of the summary table key figures and the RPMAX summary table field.
- The key figure values of the credit records flow into 0CREDIT, and the key figure values of the debit records flow into 0DEBIT.
- 0SALES is calculated from 0DEBIT + 0CREDIT during the extraction process.
- 0BALANCE is also calculated: 0BALANCE for the previous period + 0SALES for the current period. Specific DB selections during the extraction process are therefore required in order to fill the 0BALANCE field, because the records from previous periods also need to be accessed. In BI, the 0BALANCE InfoObject is subject to the *last value* exception aggregation (ref. 0FISCPER).

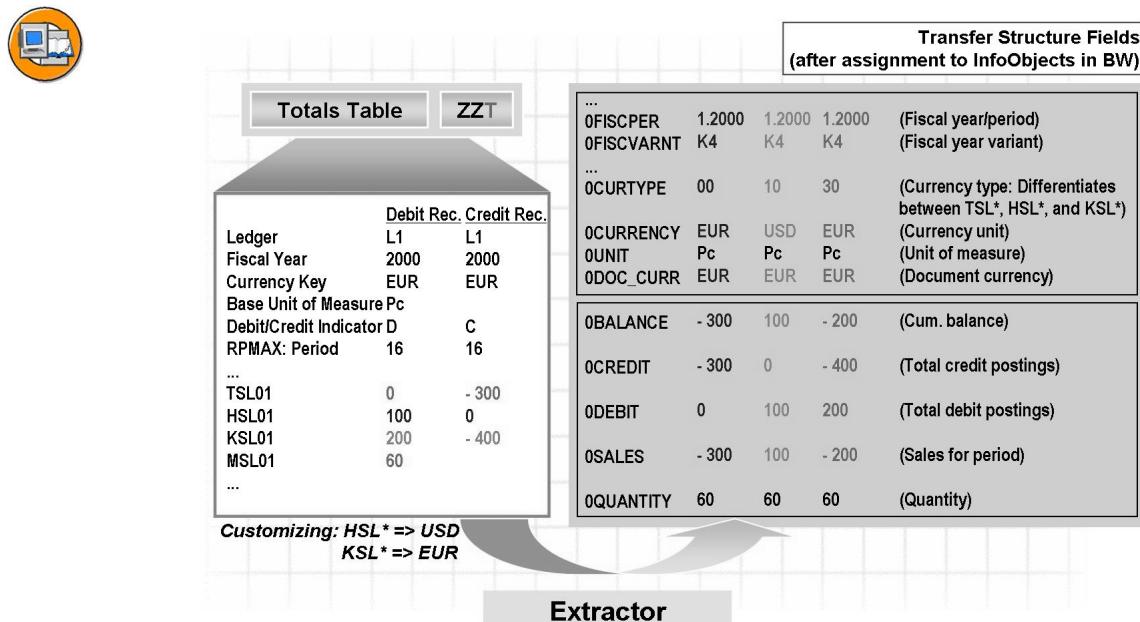


Figure 229: Extractor: Handling Currencies and Quantities

This example shows how the extractor uses the currencies and quantity key figures from an FI-SL summary table.

Currencies:

- In 0DOC_CURR the currency key of the transaction currency always flows from the FI-SL summary table.
- The 0CURTYPE characteristic differentiates between the different currency types (document currency, company code currency, and so on).
- The currency unit (DEM, USD, EUR) of the specified currency type flows into 0CURRENCY.

Quantities:

- The quantity key figures MSL01 to MSL16 are mapped to the 0QUANTITY key figure. The summary table base unit of measure is displayed in the 0UNIT InfoObject. Warning! When you create a query, you must restrict the 0CURTYPE characteristic to one characteristic value in the query definition if you want to report on quantities (0UNIT), because at the present time the quantities are transferred many times over (see example above.)
- The additional quantities (ASL01 to ASL16 in the summary table) are not transferred into BI at the present time.

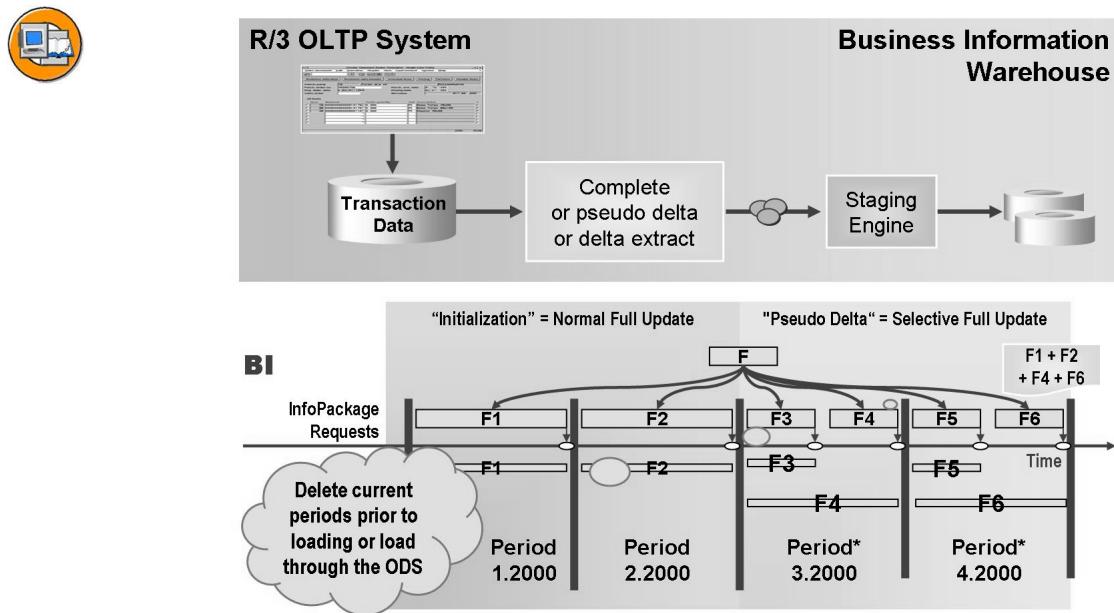


Figure 230: Extracting FI-SL Transaction Data

The update modes *Full Update* and *Delta Update* are available for the transaction data DataSources 3FI_SL_*. However, the delta process is available only in the standard R/3 Release 4.70 (R/3 Enterprise) and above. In R/3 Releases 4.0B to 4.6D, you can use SAP Note 328197 to implement the delta method. The delta process only supplies "actual" data.

If the delta process is not available, or if it would take too long to implement the note, you can set up a pseudo delta process as follows:

- We recommend you use a full update in the InfoCube(s) for all closed posting periods.
- You can check whether a posting period is open or closed in the OLTP IMG under *Financial Accounting → Special Ledger → Actual Posting → Posting Periods* (shows open posting periods).
- For all posting periods that are "still open", a full update in the DataStore Object is recommended. These records should also be updated in the InfoCube(s).
- After the transaction data records for posting periods that are "still open" in the OLTP have been changed, the records are transferred again to the DataStore Object. When you use the delta function in the DataStore Object, only new and changed records from the DataStore Object (the delta) can be updated in the InfoCube(s).



(in the BI IMG of the OLTP system)

You can use a generic set hierarchy DataSource to transfer set hierarchies for characteristics from R/3 FI-SL totals tables to BI.

Set hierarchies for characteristic "Business Area" (RBUSA):

Extracting the set hierarchy using generated DataSource 4R_GSBER_0000_HIER:

Figure 231: Setting Sp a Set Hierarchy DataSource

You use transaction /NSBIW to call the BI IMG for the OLTP system:

Settings for Application-Specific DataSources → Financial Accounting – Special Ledger → Generate DataSources for Set Hierarchies

Prerequisites: You have already created set hierarchies in the SAP system (set class 0000 = general sets).



Hint: To create a set hierarchy, choose *Accounting → Financial Accounting → Special Ledger → Tools → Set Maintenance → Sets → Create* in the SAP system.

Activities for generating a DataSource for a set hierarchy:

1. Enter the name of the table that the set uses (for example, an FI-SL summary table GLFUNCT) and select the table field (for example, *RBUSA Business Area*) for the set.
2. If the characteristic that the hierarchy is based on is a compound characteristic in BI and if you want to transfer this compounding information from the source system into BI, select the *BI InfoObject (Characteristic) is Compounded* option. Enter the name of the data element to which the hierarchy is compounded. When the hierarchy is loaded, the compounding information is included in BI. If you want to extract the FI-SL set of a characteristic that is compounded in BI, you have to run the compounding process in a customer exit. In transaction CMOD, create a project for SAP enhancement RSAP0001 and program EXIT_SAPLRSAP_004 (ZXRSAU04).
3. Run the program to generate the DataSource for the set hierarchy.
4. The new DataSource is called 4R_GSBER_0000_HIER. GSBER is the data element of the GLFUNCT-RBUSA field and 0000 is the set class.



Lesson Summary

You should now be able to:

- Position FI-SL in the SAP system
- Describe FI-SL data structures
- Generate FI-SL DataSources
- Explain the particular methods used to extract data from FI-SL
- Create generic DataSources for set hierarchies



Unit Summary

You should now be able to:

- Design and define DataSources in CO-PA
- Build an InfoCube and maintain transformation rules
- Manage CO-PA extraction
- Use CO-PA tools for BI
- Position FI-SL in the SAP system
- Describe FI-SL data structures
- Generate FI-SL DataSources
- Explain the particular methods used to extract data from FI-SL
- Create generic DataSources for set hierarchies



Course Summary

You should now be able to:

- Describe the entire process of data extraction from SAP components
- Explain the data flow within BI

Feedback

SAP AG has made every effort in the preparation of this course to ensure the accuracy and completeness of the materials. If you have any corrections or suggestions for improvement, please record them in the appropriate place in the course evaluation.