SAP NetWeaver
How-To Guide

How To Use Data Services II - Data Quality For Experts

Applicable Releases:

SAP NetWeaver 7.0

IT Practice:

Business Information Management

IT Scenario:

Enterprise Data Warehousing

Version 1.0

November 2008

Document History

Document Version	Description
1.00	First official release of this guide

Typographic Conventions

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

Icons

Icon	Description
	Caution
	Note or Important
	Example
	Recommendation or Tip

Table of Contents

1. Business Scenario

Data Quality – step up to the next level.

After introducing into basic Data Quality measures, like profiling, plausibility checks, pattern and string matching, there are many scenarios that require more sophisticated measures and tools. Some scenarios could be

- The validity of a customer's or a person's address globally or in within a specific country
- Elimination of duplicate records within a data set based on customer-defined criteria (rules). This can be duplicate entries for customers and persons, but also for example for various types of Master Data, as Material, Projects, Cost Centers, Accounts, and others.
- Calculation of checksums, averages for key figures, but also the counting of records during the staging process, i.e. do I have the same number of records in the source and the target

Features like Address and Data Cleansing, Matching and Auditing can alleviate most of the issues in this area.

Similar to the topics already covered in the first publication, Data Cleansing, Matching and Auditing can be realized in SAP BI only with a huge effort or not at all. Many features like dictionaries, parsing rules, etc. would have to be implemented in ABAP.

Data Services provides these features out-of-the-box with inbuilt / delivered dictionaries, matching and parsing rules. In addition, the user can define own dictionaries and custom-defined rules to tailor / enhance Data Services to their requirements.

After processing the data set in Data Services, it can easily be incorporated into the SAP BI staging process, thus bolstering the level of Data Quality in the SAP BI system to a great extent.

2. Background Information

2.1 Introduction to Data Services features used

The features we want to introduce and use in this document are:

1. **Data / Address Cleansing**
   This tool parses, cleanses and standardizes data such as names/addresses, emails, phone numbers, Social Security Numbers, and dates into individual components. It manages international data for over 190 countries and reads and writes Unicode data. It improves integrity of data to identify matches and ultimately create a single customer view
   The *Data Cleanse* Transform uses rule-based parsing, identifying and isolating specific parts of mixed data, and standardizes your data based on information stored in the parsing dictionary, business rules defined in the rule file, and expressions defined in a pattern file. Address Cleansing is particular a version of Data Cleansing, whereby Data Services provides pre-customized Transforms for various countries. The basic proceeding of the cleansing is the same for both methods.

   You can use *Data Cleanse* to assign gender codes and prenames, split records with dual names into individual records, create personalized greetings, and generate standards used in

the match process *Data Cleanse* can also parse and manipulate various forms of international data, as well as operational and product data.

2. **Matching and Consolidation**
   The matching and consolidation component of Data Services solution matches and consolidates data elements based on user-defined business rules. Duplicate records can be identified and eliminated. All information of each individual customer or an entire corporation or household is consolidated, hence only the unique records are saved to the database. Since the matching logic is governed by a set of match rules that can be customized to implement a user-defined solution, you are not forced to adopt pre-established rules. This provides the flexibility in determining what is a “match” in the database.

3. **Auditing**
   The *Auditing* feature of Data Services allows collecting run time statistics about data that flows from the source to the target, hence improving the quality of the *DataFlow*. It can provide information about number of occurrences, checksums and other calculation on columns of a data set like Average, Sum calculation, etc.. Auditing can be applied to Sources, Transforms and Targets. Rules allow to define logical expressions for dependencies among multiple audit statistics within the same dataflow, e.g. that the count of rows from the source table is the equal to the rows in the target table, that certain thresholds for checksums are not exceeded, etc.. In case of deviation from a set standard it is possible to trigger notifications.

3. Prerequisites

- Software
  - Business Objects Data Services XI 3.0

    Note

    The information provided should also be applicable to prior or future releases of Data Services, though with changes in the realization.
- Hardware
  - No particular hardware needed (apart from the standard specifications for SAP NetWeaver BI and Data Services)

Provide information about:

- Relevant SAP Notes
  - None
- Additional background/starting documentation (also provide a link)
  - For SAP BI - help.sap.com (SAP Solutions → SAP NetWeaver → SAP NetWeaver 7.0 → Functional View → SAP NetWeaver by Key Capabilities → Information Integration → Business Intelligence)
  - For Data Services - help.sap.com (Business Objects → choose product and release)
    - Data Services Designer Guide
    - Data Services Reference Guide

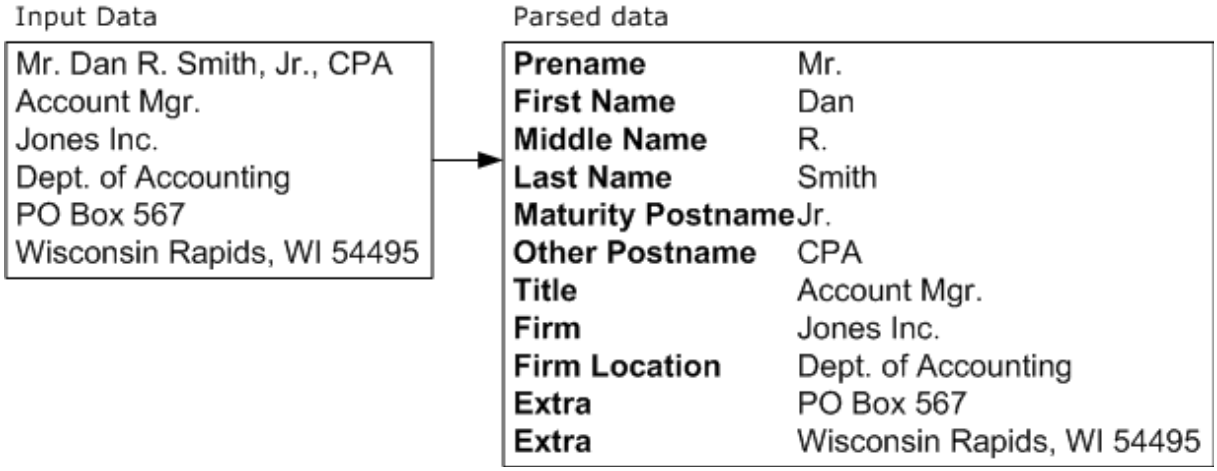
    - Additional information and implementation assistance can be found in the various communities like SDN (SAP NetWeaver BI and Business Objects products) and DIAMOND (Business Objects technical community)

- Required/recommended expertise or prior knowledge
    - SAP BI – Intermediate EDW knowledge
    - Data Services - Basic knowledge

4. Step-by-Step Procedure

4.1 Data Cleansing

Data Cleansing is analyzing / parsing a record, and populating defined output fields based on specific rules. The rules for the parsing and the output fields are either provided by Data Services or can be defined by the customer to satisfy specific requirements. This process standardizes the data and enables a higher degree of consistency within the data. It translates data in different formats into a common one.



The standard *Data Cleanse* Transforms of Data Services can even assign a gender depending on the provided name, create personalized greetings in different styles, split a single record into separate one (in case for example multiple names are contained in one record), etc.

Data Cleansing prepares the data also for a potential Matching, since the confidence in the Matching process is increased by standardized data. The *Match* Transform itself does not perform any standardization of the data. The *Data Cleanse* Transform can generate given name match standards, or potential matching words. For example, *Data Cleanse* can tell you that Patrick and Patricia are potential matches for the name Pat. Match standards can help you overcome two types of matching problems: alternate spellings (Catherine and Katherine) and nicknames (Pat and Patrick).

For components other than person and firm data, you can use *Universal Data Cleanse*, and other Transforms and Functions available in Data Services, such as the *search_replace* Function and *User-Defined* Transform to standardize the data before matching.

Data Cleanse can parse data that is outside of the range of name, title, address, etc.. With the *user-defined pattern matching (UDPM)* feature, *Data Cleanse* can parse a wide variety of data such as:

- account numbers
- part numbers
- purchase orders
- invoice numbers
- VINs (vehicle identification numbers)
- driver license numbers

In other words, *Data Cleanse* can parse any kind of number or alphanumeric field for which you can define a pattern. For details on how to use a UPDM, please refer to the Appendix of the Data Services Reference Guide.

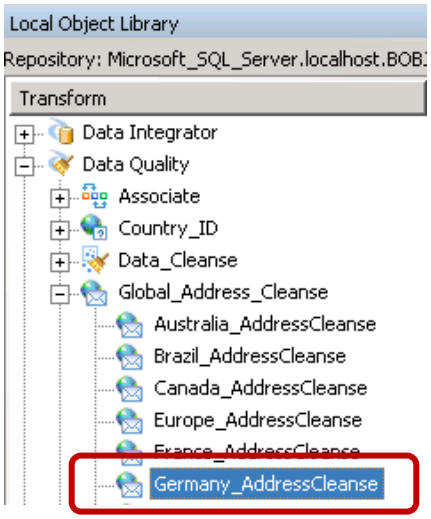
For even more complex scenario, where the complete cleansing process has to be configured, the *Universal Data Cleanse* feature could be used to cleanse the data. With this feature, a custom-defined dictionary, rules, output categories and fields, and classifications are used to perform the data cleansing.

Data Cleansing is based on a single basis Transform, which is either delivered, pre-configured for different purposes (countries, company scenarios, etc.) or can be customized to cater for specific requirements.

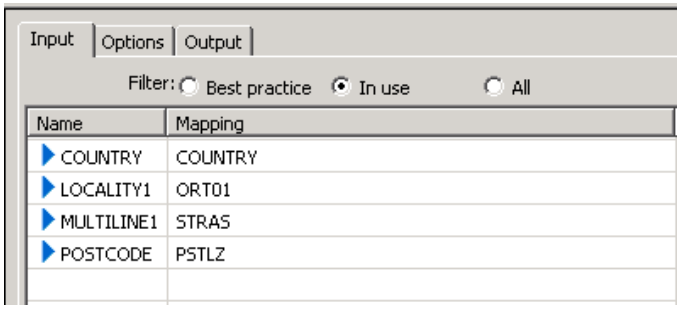
For the ease of comprehensiveness, we are using the same source data as in the first publication (Non-SAP customer data), Hence, we are using the delivered *Address Cleansing* Transform for the processing of our customer data.

The eventual *DataFlow* could look like the screen shot depicted underneath.

1. Open Data Services and define a Job and a *DataFlow*.
2. Within the *DataFlow* drag your data source to the canvas. In our example, we are using an EXCEL file with customer data.
3. Choose the *Transform* tab strip, and pick from the *Data Quality* → *Global_Adress_Cleanse* folder the *Germany_AddressCleanse* Transform. Drag it to the canvas. Connect the data source and the *AddressCleanse* Transform.



4. Double-Click on the *AddressCleanse* Transform. Choose the *Input* tab strip, and the fields that you want to use in the Address Cleansing process, most suitable fields which contain address information. We have chosen the fields *COUNTRY, PSTLZ, ORT01 and STRAS* as input fields. For detailed information about which are the most suitable fields, please check the *Designer* and *Reference Guide*.

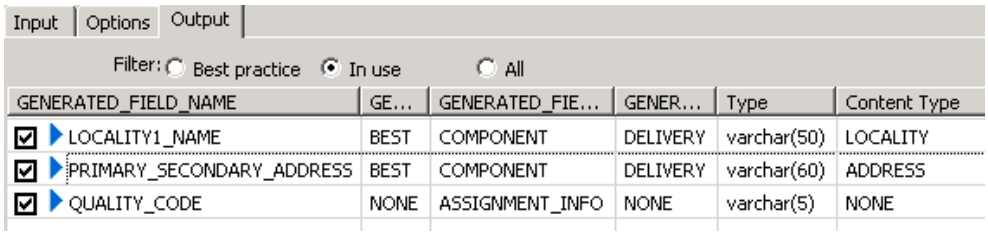


5. In general, changes in the *Options* tab strip are not required. The *Germany_AddressCleanse* Transform is particular parameterized to cater for German requirements. If needed, the settings can be changed to reflect particular customer requirements.

6. Switch to the *Output* tab strip to define the fields that should be passed on. In addition to source fields, which should be transferred from the source to eventual target in the *DataFlow*, we selected the following fields

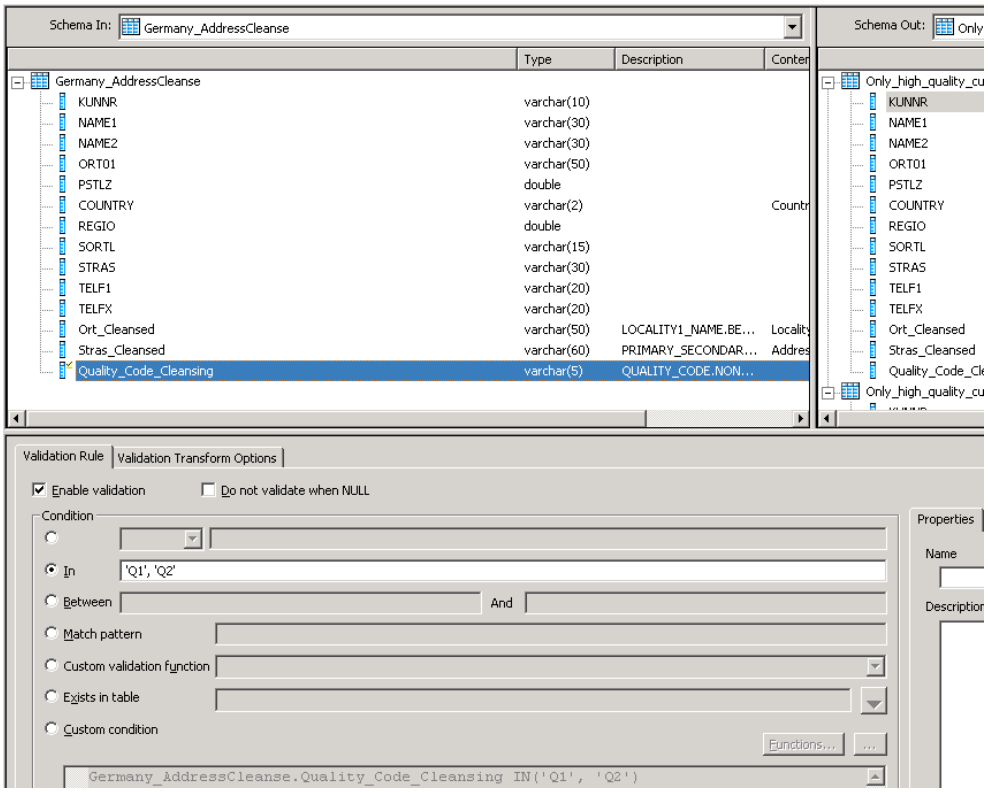
QUALITY_CODE	This field indicates the degree of the cleansing quality for a particular record.
LOCALITY1_NAME	This field contains the standardized value for the city information of the address (field *ORT01*).
PRIMARY_SECONDARY_ADDRESS	This field contains the standardized value for the street information of the address (field *STRAS*).

7. To provide a more meaningful description, you can rename the selected fields in the *Out Schema* (via the *Properties* dialog of the respective field)



8. Choose the *Transform* tab strip, and pick from the *Platform* folder the *Validation* Transform. Drag it to the canvas. Connect the *AddressCleanse* Transform and the *Validation* Transform, You can rename it to provide a more meaningful name (in our case we named it *Only_high_quality_customers* to indicate that we are forwarding only records of customers cleansed with a high *Quality Code*)

9. Double-Click on the *Validation* Transform. Define which level of quality is desired to pass on / or deny records. For the example we define a *Quality Code* of 'Q1' or 'Q2' as criteria for the records to be passed as valid. To do so, mark the field containing the Quality Code (in our case *Quality_Code_Cleansing*) and define the respective validation rule.



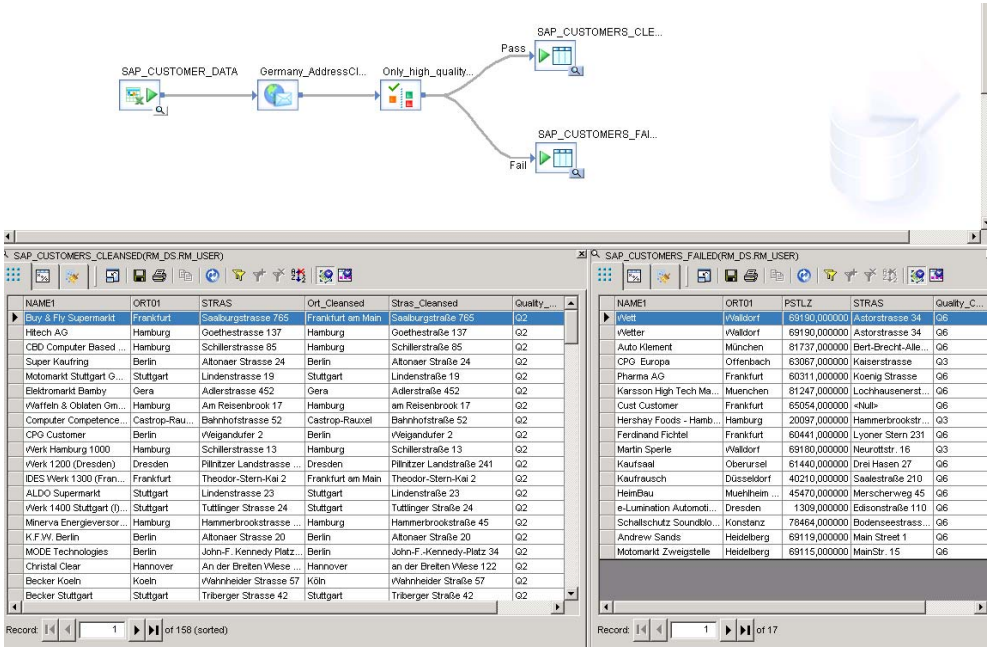
10. [Optional: If you want reduce the number of fields in the eventual target, you can add a *Query* Transform to select the respective fields. An example could be to use the cleansed / standardized fields instead of the original source fields for the city and street information of the address. This would eliminate various spellings of them for the source data]

11. Choose the targets for the valid and invalid records. This can be a table, a file or a template table. Connect the *Validation* Transform and the targets.

12. Validate your *DataFlow* by using either the menu entry Validation → Validate → Current View / All objects in view or use the respective icons ()

13. If your Job and *DataFlow* is correct, you can save and execute it. Position the cursor on the Job name, and select from the context menu *Execute*.

14. After the job has completed successfully, you can check the result of the cleansing by returning to the *DataFlow* display. After using the View Data option for the valid and invalid targets, we detect that 15 records have not been cleansed with the quality we defined ('Q1' or 'Q2') (right hand side of the screen shot underneath shows the invalid records).



4.2 Matching

Matching is based on the custom-defined business rules. The *Match* Transform allows to eliminate duplicate records, and sends matching and unique records on to the next Transform in the *DataFlow*. For best results, the data in which you are attempting to find matches should be cleansed. Hence, you will place your *Match* and related *Transforms* after *Cleansing* Transforms.

The *Match* Transform is only one tool, albeit the most important one to use in the matching strategy. For more information about matching concepts and other Transforms available to achieve a specific result you are looking for, see the *Match* section of the Data Services Designer Guide.

There are pre-defined matching strategies for simple, consumer and corporate match scenarios delivered with the product.

The main components of matching are:

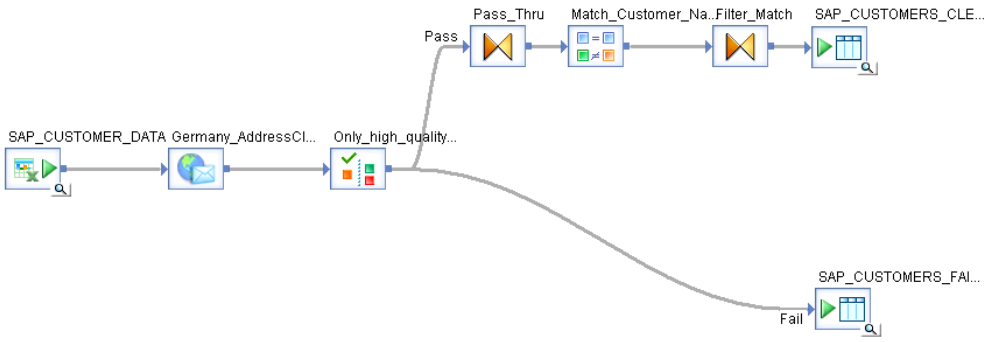
- **Match sets**
  A match sets is represented by a *Match* Transform in the dataflow. It defines how the *Match* Transform matches records, and consists of break groups, match criteria and prioritization.
- **Match levels**
  A match level defines the level on which the matching occurs, i.e. an individual, family, resident, firm, etc. If multiple levels are used, the levels define a hierarchy in respect of being stricter on every next level, for example from resident to family to the individual.
- **Match criteria**
  The match criteria defines the fields the matching is performed on. It can be specified how close to exact the data needs to be for being considered as a match.

- **Break key (group)**
  Break Groups (Break Keys) define manageable groups of data to compare. The match set compares the data in the records within each *Break Group* only, not across the groups. Making the correct selections can save valuable processing time by preventing widely divergent data from being compared. *Break Groups* are especially important when you deal with large amounts of data, because the size of the *Break Groups* can affect processing time. For example, when you match to find duplicate addresses, base the *Break Group* on the postcode, city, or state to create groups with the most likely matches.
  Another means for performance optimization is the candidate selection to add a smaller data set in an existing bigger one.

Similar to Data Cleansing, matching is based on a single basis Transform, which is either delivered, pre-configured for different scenarios corresponding to different match strategies or can be customized to cater for specific requirements. For the creation of custom-defined *Match* Transforms in the dataflow, a *Match Wizard* is available. It allows the definition of the main parameters. They can be adjusted (or extended) afterwards.

> **Note**
>
> You should filter out empty records before matching. This improves the performance. Use a *Case* Transform to route records to a different path or a *Query* Transform to filter or block records.

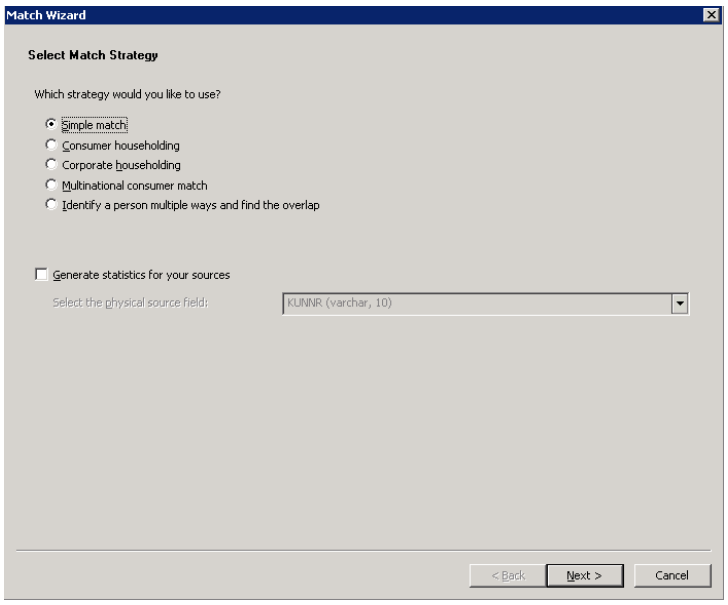
For our example, we are enhancing the existing *DataFlow* containing the cleansed data with the Transforms needed for the Matching. Since we want to have only a single entry for every corporation, we determine which customers have multiple entries based on their name (regardless of their address). The resulting *DataFlow* could look like the screen shot depicted underneath.



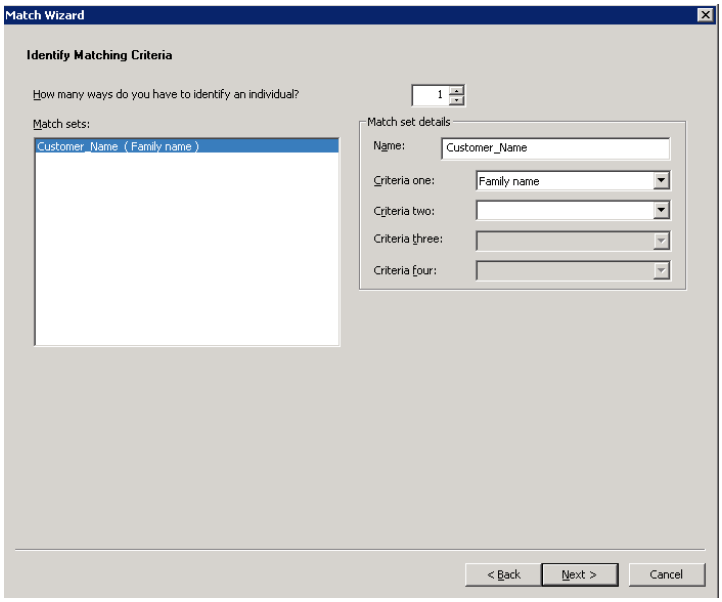
1. Open the *DataFlow* (or replicate it to keep the original one), you have created in the previous section. Delete the connection between the *Pass* output of the *Validation* Transform and the target for the valid customers.
2. Drag a simple *Query* Transform (1:1 mapping of all input fields to the output) to the canvas, and connect to the *Pass* output of the *Validation* Transform.

> **Note**
>
> The *Match* Transform cannot be attached directly to a *Validation* Transform. A *Query* Transform has to be introduced to pass-on the fields.

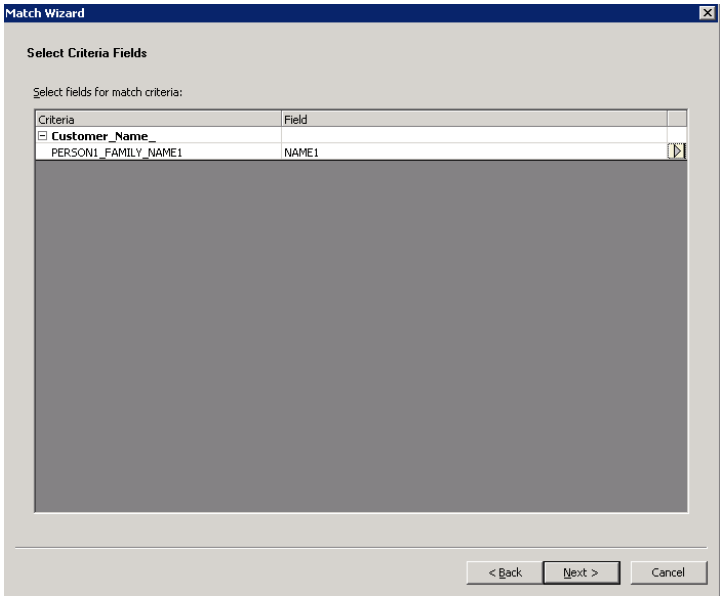
3. Position the cursor on the just added *Query* Transform (in our case named *Pass_Thru*) and choose the option *Run Match Wizard → Pass* from the context menu.
4. Choose the option *Simple match* or any other suitable *Match Strategy*. We have selected *Simple*, since we want to match the customers only based on their name. Press the *Next* button.



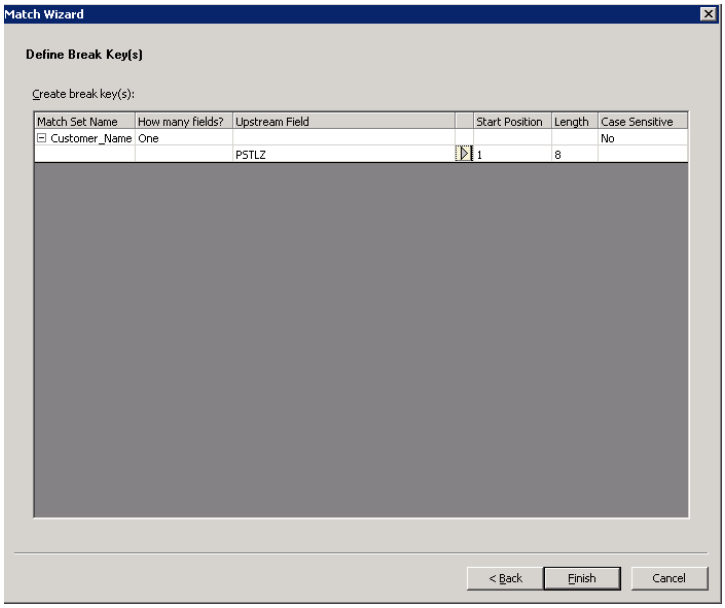
5. The creation of one *Match Set* leads to one *Match* Transform in the *DataFlow*. Define one *Match Set* based on a single field. To achieve this, we pick the single *Criteria Family_Name*. If the *Match* Transform should have a specific name, specify the name in the *Name* field. We named the *Match Set Customer_Name*. Press the *Next* button.



6. Select the field(s) which should be used for the respective *Match Criteria*. Since we use the customer name as *Match Criteria*, the field *NAME1* has been picked from the drop-down list. Press the *Next* button.



7. Define a break key (group). One or multiple fields with flexible offsets (start position and length) can be picked from the drop-down list. We are using a single field, the field *PSTLZ* which breaks the data set to be matched in groups per postal code. Press the *Finish* button. Afterwards, the *Match* Transform is automatically connected to the *Query* Transform in the *DataFlow*.



8. If you want to adjust certain settings for the *Match* Transform, choose the entry *Match Editor* from the context menu or double-click on the Transform itself, and change the respective settings. Most relevant settings might be the settings *Match score, No Match score and Use in weighted score if greater than.*

Important

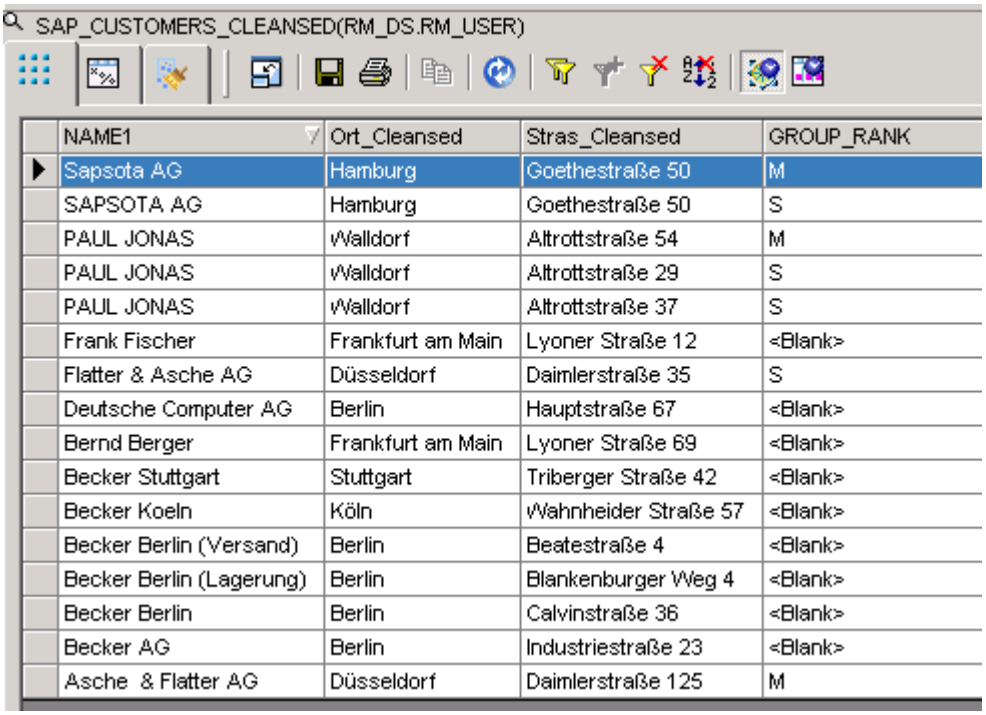
If the *Match Score* is adjusted, the *Use in weighted score if greater than* has to be adjusted to be between the *Match* and the *No Match* score.

9. If the execution of the Match Transform was executed at this point in time, the output could look like the underneath screen shot. Some of the fields, that can be used to assess the match result are:

GROUP_NUMBER This field indicates the group wherein matching records are collected in.

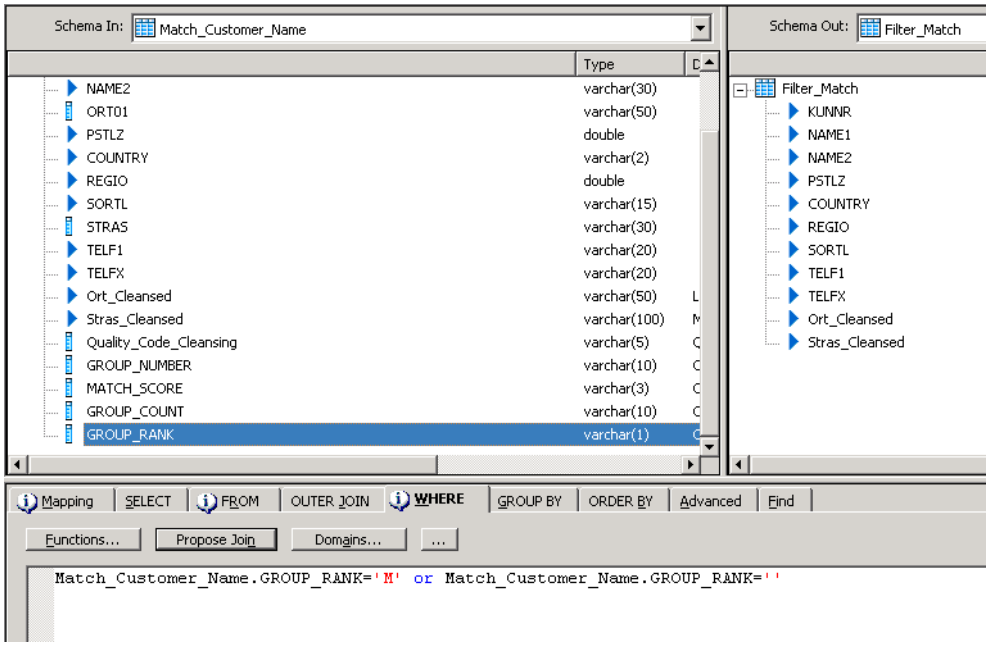
GROUP_RANK This field defines the role of a record within a group. The value 'M' defines the master record, whereas the value 'S' defines the slaves (or records that match the master / leading record).

MATCH_SCORE This field contains the degree of matching a record has in relation to the master / leading record.



NAME1	Ort_Cleansed	Stras_Cleansed	GROUP_RANK
Sapsota AG	Hamburg	Goethestraße 50	M
SAPSOTA AG	Hamburg	Goethestraße 50	S
PAUL JONAS	Walldorf	Altrottstraße 54	M
PAUL JONAS	Walldorf	Altrottstraße 29	S
PAUL JONAS	Walldorf	Altrottstraße 37	S
Frank Fischer	Frankfurt am Main	Lyoner Straße 12	<Blank>
Flatter & Asche AG	Düsseldorf	Daimlerstraße 35	S
Deutsche Computer AG	Berlin	Hauptstraße 67	<Blank>
Bernd Berger	Frankfurt am Main	Lyoner Straße 69	<Blank>
Becker Stuttgart	Stuttgart	Triberger Straße 42	<Blank>
Becker Koeln	Köln	Wahnheider Straße 57	<Blank>
Becker Berlin (Versand)	Berlin	Beatestraße 4	<Blank>
Becker Berlin (Lagerung)	Berlin	Blankenburger Weg 4	<Blank>
Becker Berlin	Berlin	Calvinstraße 36	<Blank>
Becker AG	Berlin	Industriestraße 23	<Blank>
Asche & Flatter AG	Düsseldorf	Daimlerstraße 125	M

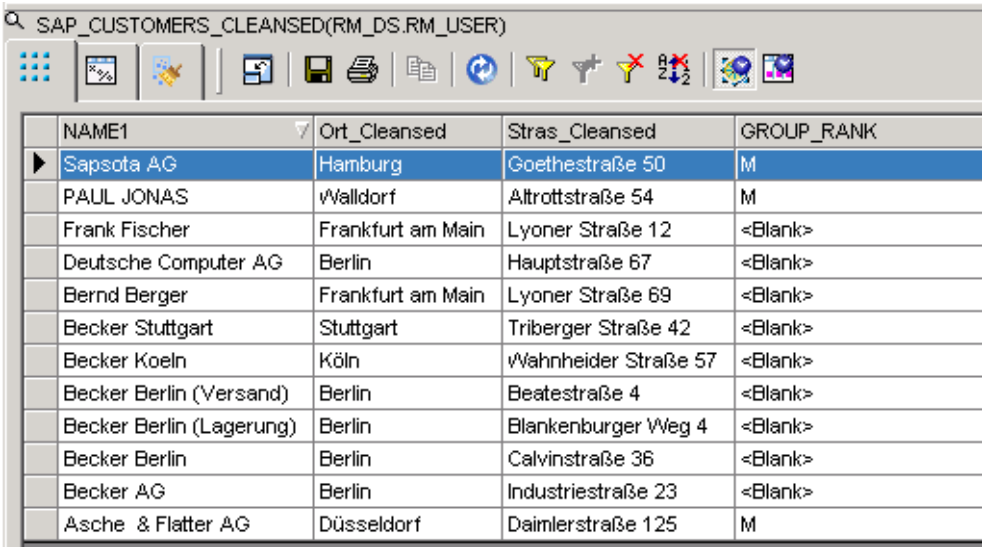
10. In order to forward only a single record of the matching records (and all of the unique records), we filter only the records which have a group rank with the value 'M' or BLANK. To achieve this result, a *Query* Transform is introduced in the *DataFlow*. Connect the *Query* Transform with the target for the valid customers. Within the Query Transform, define a WHERE clause that selects only records with the value 'M' or BLANK.



11. Validate your *DataFlow* by using either the menu entry Validation → Validate → Current View / All objects in view or use the respective icons ()

12. If your Job and *DataFlow* is correct, you can save and execute it. Position the cursor on the Job name, and select from the context menu *Execute*.

13. After the job has completed successfully, you can check the result of the matching by returning to the *DataFlow* display. Inconsistent customer records have already been eliminated with the *Data Cleansing.* The matching found duplicates for three entries (customers 'SAPSOTA AG', 'PAUL JONAS' and 'ASCHE & FLATTER AG'), resulting in the elimination of a total of four records (since one entry has two duplicates). After using the View Data option for the valid target, we detect that the duplicate records are eliminated (see also value for column *GROUP_RANK* does not contain any 'S' values).



NAME1	Ort_Cleansed	Stras_Cleansed	GROUP_RANK
Sapsota AG	Hamburg	Goethestraße 50	M
PAUL JONAS	Walldorf	Altrottstraße 54	M
Frank Fischer	Frankfurt am Main	Lyoner Straße 12	<Blank>
Deutsche Computer AG	Berlin	Hauptstraße 67	<Blank>
Bernd Berger	Frankfurt am Main	Lyoner Straße 69	<Blank>
Becker Stuttgart	Stuttgart	Triberger Straße 42	<Blank>
Becker Koeln	Köln	Wahnheider Straße 57	<Blank>
Becker Berlin (Versand)	Berlin	Beatestraße 4	<Blank>
Becker Berlin (Lagerung)	Berlin	Blankenburger Weg 4	<Blank>
Becker Berlin	Berlin	Calvinstraße 36	<Blank>
Becker AG	Berlin	Industriestraße 23	<Blank>
Asche & Flatter AG	Düsseldorf	Daimlerstraße 125	M

4.3 Auditing

The definition of *Auditing* can be started at all relevant points for a particular dataflow:

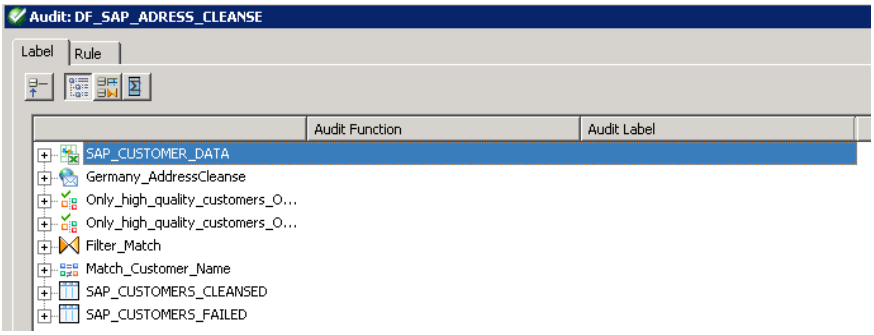
- From the *DataFlows* tab strip of the object library, right-click on a *DataFlow* name and select the *Auditing* option from the context menu.
- In the workspace, right-click on a *DataFlow* icon and select the *Auditing* option from the context menu.
- When a *DataFlow* is open in the workspace, click the *Audit* icon in the toolbar (),

For our example, we want to track that the number of records in the initial source and the eventual target is the same. If not, we know that one or multiple of the following situations might have occurred:

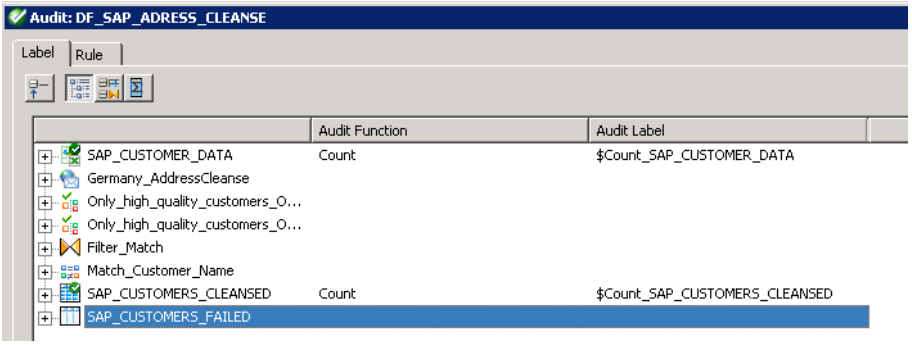
- Some records have been marked as invalid in the *Cleansing* Transform
- There existed some duplicated records in the data set which have been identified and eliminated by *Matching*
- Another cause (potentially an error) in one of the Transforms has reduced the number of records

In case of a deviation, an administrator should be notified to analyze the cause, and initiate actions if required.

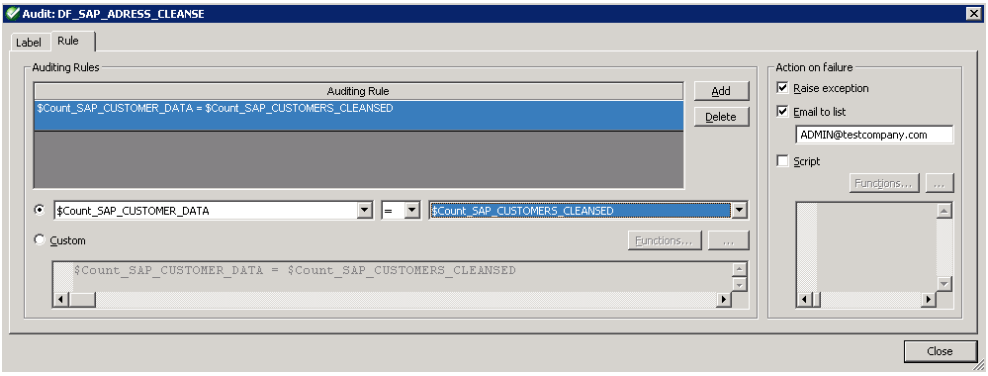
1. Open the *DataFlow* you have created in the previous section.
2. Click the *Audit* icon in the toolbar () to open the *Audit* dialog. Alternatively, use one of the methods described above.



3. Define the points in the *DataFlow* where you want to use the *Auditing* feature. You do so, by positioning on the respective objects or fields and selecting the desired entry from the context menu. We picked the entry *Count* for the source (*SAP_CUSTOMER_DATA*) and the target for the valid customers (*SAP_CUSTOMERS_CLEANSED*). The dialog shows the details of the *Auditing* for the objects.

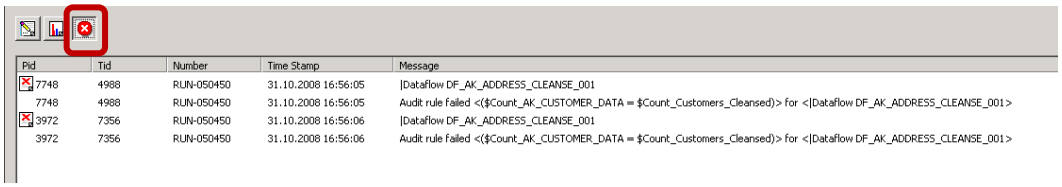


4. Switch to the *Rule* tab strip to define a rule and an action in case of a failure of the rule. In our case we picked the Count fields / Audit levels for the source and the target from the dropdown list. We defined that both values have to be equal (operator '='). In case of failure, we define as action to raise an exception (which will cause the job to be stopped), and to send an Email to the user 'ADMIN@testcompany.com'.



5. Validate your *DataFlow* by using either the menu entry Validation → Validate → Current View / All objects in view or use the respective icons ()
6. If your Job and *DataFlow* is correct, you can save and execute it. Position the cursor on the Job name, and select from the context menu *Execute*.

If the *Auditing Rule* is failing, i.e. we have a deviation of the number of records in the source and in the target; entries in the error part of the job log are written. In order to see the entries, press the error button in the job log ().



In addition, the administrator will receive a mail with a notification about the failure of the *Auditing Rule.*

4.4 Loading data from Data Services into SAP BI

In order to include the Data Quality measures into the SAP BI staging process, we have to load the cleansed and matched data into the SAP BI system. For details about how to connect an SAP BI system to Data Services, and how to load the data from Data Services to the SAP BI system, please refer to the first publication of your series (How To Data Services II - Data Quality Made Easy).

For a rough understanding, we are sketching underneath the process.

1. To import the SAP BI structures, open the *Datastore* tab strip in the Object Library. Search for the Datastore that you have created for the SAP BI system as target. Position the cursor on the Datastore name and choose *Open* from the context menu. Depending on the structures you want to use, expand the Master InfoSources or Transaction InfoSources tree. Find your *InfoSource*, and open its subtree. Position the cursor on the DataSource name, and use the option *Import* from the context menu. Afterwards, the DataSource will be available in the Object Library for your SAP BI system.

2. Create a Data Services Job which uses the SAP BI *Datastore* structure as target.
3. Open the Data Services Management Console by choosing the corresponding menu entry from the *Tools* menu.
4. Switch to the *Batch Job Configuration* tab strip and find your Job you want to schedule. Choose the option *Export Execution Command.*
5. Provide a File Name for the batch file. Leave the other settings on the default values and press the *Export* button.

   Note

   Since the maximum length of the *file name* entry field in the InfoPackage is limited to 44 characters, your file name entered must not exceed 40 characters (44 characters minus 4 characters for the extension *.bat*)
6. Before you switch to the SAP BI system, check that the RFC server is still running.
7. Switch to the SAP BI system you want to load the data to. Open the *Data Warehousing Workbench and* find your InfoSource / DataSource. Create an InfoPackage for the DataSource. Switch to the *3rd Party Selection* tab strip. Press the *Refresh Sel. Fields* button to display the input fields.
8. Enter the File Name that you have provided in the creation of the batch file.
9. You can now schedule the load. If you schedule the execution at a future point in time, please check that the RFC server is running at the time of execution. After the execution of the InfoPackage, you can check in the monitor the status of the load.

www.sdn.sap.com/irj/sdn/howtoguides