CHAPTER

1

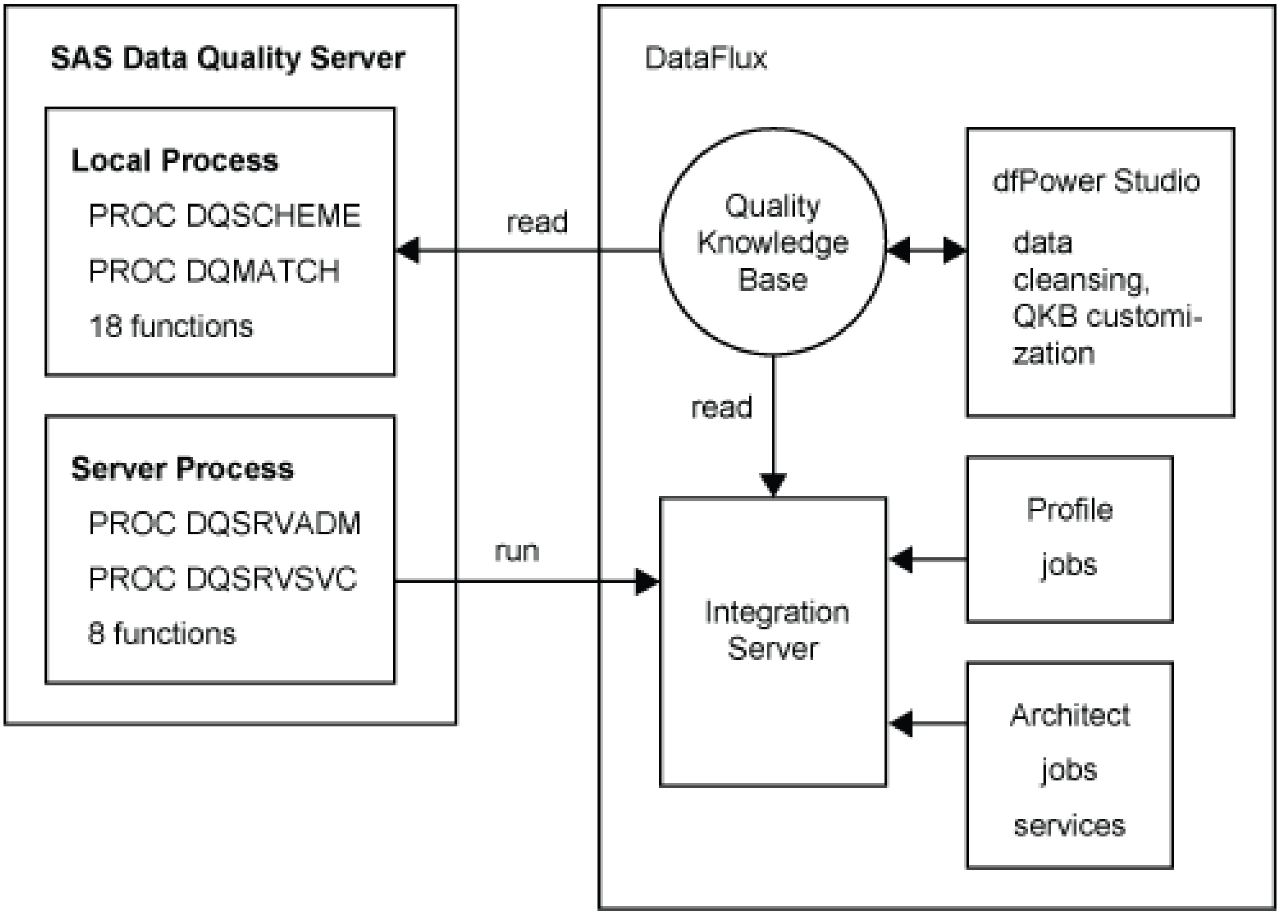
Using the SAS Data Quality Server Software

Overview

The SAS Data Quality Server software provides a Quality Knowledge Base, along with SAS language elements that enable you to analyze, transform, and standardize your data. By cleansing your data, you increase the quality and value of the information that you derive from your data.

The language elements in the SAS Data Quality Server software can be separated into two functional groups. As shown in the following diagram, one group cleanses data in SAS. The other group runs data cleansing jobs and services on Integration Servers from DataFlux (a SAS company).

Figure 1.1 Functional Overview of the SAS Data Quality Server Software



The language elements in the Local Process group read data definitions out of the Quality Knowledge Base to, for example, create match codes, apply schemes, or parse text. The language elements in the Server Process group start and stop jobs and services and manage log entries on DataFlux Integration Servers.

The DataFlux Integration Servers and the related dfPower Profile and dfPower Architect applications are made available with the SAS Data Quality Server software in various software bundles.

You can add DataFlux software after purchasing the SAS Data Quality Server software.

DataFlux Integration Servers that are purchased for use with SAS are restricted so that jobs and services can be executed only by SAS programs.

About the Quality Knowledge Base

The Quality Knowledge Base contains data quality *definitions* for a number of *locales*. Definitions specify how categories of data are processed. For example, the match definition NAME defines how match codes are created for the names of individuals. The data definitions in the Quality Knowledge Base are redefined in each locale.

Locales provide data definitions for a national language and a geographical region. For example, the locale ENUSA reflects the English language as it is used in the United States.

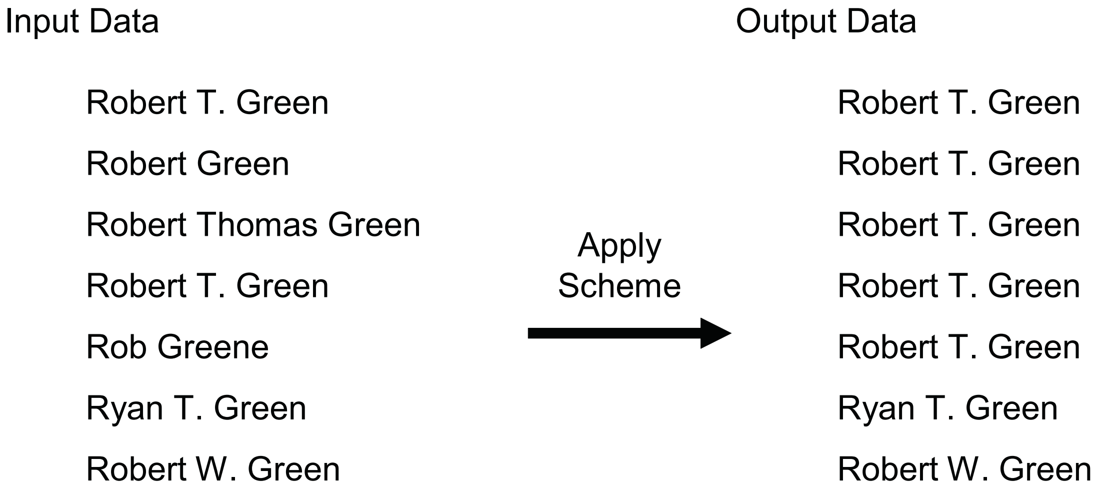
In your data analysis and cleansing programs, and in your DataFlux jobs, and services, you select the locales that apply to your data. Those locales are read into your process for reference during analysis, transformation, and standardization.

To obtain online Help for your Quality Knowledge Base, open the dfPower Customize application and select **Help ▶ QKB Help**.

About the Local Process Group

The local process group of SAS language elements provides data cleansing capabilities within SAS processes. The group consists of the procedures DQSCHEME and DQMATCH, 18 functions, one CALL routine, 2 system options, and several AUTOCALL macros.

The DQSCHEME procedure enables you to create and apply schemes that transform similar data values into the single-most-common value, as shown in the following diagram.



Input Data	Output Data
Robert T. Green	Robert T. Green
Robert Green	Robert T. Green
Robert Thomas Green	Robert T. Green
Robert T. Green	Robert T. Green
Rob Greene	Robert T. Green
Ryan T. Green	Ryan T. Green
Robert W. Green	Robert W. Green

PROC DQSCHEME also analyzes and reports on the quality of your data. For additional information, see Chapter 3, "The DQSCHEME Procedure," on page 31.

The DQMATCH procedure enables you to generate match codes as a basis for standardization or transformation. Match codes are character representations of data values. You generate match codes based on a definition and sensitivity value. The match codes reflect the relative similarity of data values. Values that generate the same match codes are candidates for transformation or standardization. To learn more about PROC DQMATCH, see Chapter 2, "The DQMATCH Procedure," on page 19.

The functions and CALL routines in the local process group provide the following capabilities:

- □ parse a text string and return the string with tokens that identify the elements in the string
- □ return a token from a parsed string
- □ create match codes for strings or tokens
- □ standardize strings
- □ apply schemes
- □ analyze patterns
- □ return the name of the data definition that fits the value
- □ return the name of the locale that best fits the data
- □ return a list of definitions in a locale

For links to specific functions, see “Functions Listed by Category” on page 57.

About the Server Process Group

The server process group of language elements enable you to run DataFlux jobs and services on DataFlux Integration Servers. You can also administer log files on Integration Servers.

Use the DQSRVSVC procedure to run services that were created with the dfPower Architect software. These services rapidly process and return small amounts of data. Rapid response enables real-time data cleansing at the point of data entry. For more information, see Chapter 5, “The DQSRVSVC Procedure,” on page 47.

Use the DQSRVADM procedure to return job log information. See Chapter 4, “The DQSRVADM Procedure,” on page 43.

Use the functions in the server process group to run jobs, read and manage status logs, and terminate jobs and services. See “DataFlux Integration Server Functions” on page 57.

Note: The server group of procedures and functions are not available in the OpenVMS operating environment. △

About DataFlux Jobs and Services

On DataFlux Integration Servers, jobs and services fulfill separate needs. Use jobs to access larger data sets in batch mode, when a client application is not waiting for a response. Use services and small data sets in real time, when clients await a response from the server.

To create jobs and services for your Integration Servers, use the dfPower Profile and dfPower Architect applications from DataFlux (a SAS company). The dfPower Profile software enables you to create jobs that analyze the quality of your data. Profile jobs are available in two types. One type runs on individual files; the other runs on repositories. Each type of job has its own trigger function (DQSRVPROFJOBREP and DQSRVPROFJOBFILE).

The dfPower Architect software provides a graphical interface that you use to create jobs and services. You trigger the Architect jobs with the function DQSRVARCHJOB. Run Architect services with PROC DQSRVSVC.

Jobs and services generate log information on Integration Servers. You can read the server logs using the DQSRVSTATUS function or the DQSRVADM procedure. Based on the job status information that is returned in the logs, you can terminate jobs and services using DQSRVKILLJOB.

Data Quality Software in Other SAS Products

The SAS Data Quality Server software is frequently used in tandem with the dfPower Studio software from DataFlux (a SAS company). The dfPower Studio software enables you to cleanse data with or without using a DataFlux Integration Server. The dfPower Studio software includes, among other packages, the dfPower Customize software. The dfPower Customize software enables you to create and edit definitions in your Quality Knowledge Base. dfPower Studio also includes the dfPower Profile software and the dfPower Architect software, which is used to create jobs and services through a graphical user interface. For more information on dfPower Studio, see **`www.dataflux.com`**.

The SAS Data Quality Server software is bundled into enterprise versions of the SAS Intelligence Platform. In those bundles, the SAS Data Quality Server software is made available in transformation templates in the SAS Data Integration Studio software. Also, in the SAS Enterprise Guide and SAS Data Integration Studio software, the Expression Builder enables you to add data quality functions to your logical expressions.

Considerations for Installing and Updating the Software

Keep the following information in mind as you install the SAS Data Quality Server software.

The SAS Data Quality Server software is delivered with a sample Quality Knowledge Base. After you install the SAS Data Quality Server software, you need to download the latest Quality Knowledge Base, including your choice of locales, from the DataFlux Web site **`www.dataflux.com`**.

To maximize performance, download new Quality Knowledge Bases as they are made available by DataFlux. Because This generally means that you need to download a new Quality Knowledge Base immediately after you finish installing the SAS Data Quality Server software. On the DataFlux Web site, check the release notes of the latest release to determine whether your locales have been updated, or to determine whether you need any of the new locales that might have been added.

When you update your Quality Knowledge Base, you might want to install it in a new location rather than overwriting your existing Quality Knowledge Base. This decision nis particularly important if you have customized your Quality Knowledge Base in the previous release. (Customizations are made with the dfPower Customize software). If you install your updated Quality Knowledge Base in a new location, you will either need to change your setup file or reference a new setup file in your SAS programs.

If you customized your previous Quality Knowledge Base, make sure that you evaluate those changes and carry them over to your new Quality Knowledge Base.

CAUTION:

When you upgrade your Quality Knowledge Base, be sure to regenerate your existing match codes so that they will be consistent with the newly created match codes. △

Note: SAS programs that run jobs or services on DataFlux Integration Servers do not reference the setup file. △

Run Jobs and Services on DataFlux Integration Servers

Follow these steps to run jobs and services on Integration Servers from DataFlux (a SAS company):

1 In the z/OS operating environment, for DataFlux Integration Servers version 8.1.1 or newer, you are required to configure your DataFlux Integration Servers to use the Wireline protocol, as described in the *DataFlux Integration Server: User's Guide*. The Wireline protocol improves data transfer performance.

2 In the z/OS operating environment, you are also required to configure your SAS session to use the Wireline protocol, as described in "Configure Your SAS Session for Data Quality" on page 9.

3 Create jobs and services using the dfPower Profile and dfPower Architect software from DataFlux.

4 Upload the jobs to Integration Servers using the Integration Server Manager from DataFlux.

5 Create and run the SAS programs that run the jobs and services.

To run jobs and services, you do not need to load locales onto your local host. The DataFlux Integration Servers handle all interaction with your Quality Knowledge Bases.

For additional information, see these sections:

"About the Server Process Group" on page 4

"About DataFlux Jobs and Services" on page 4

Chapter 5, "The DQSRVSVC Procedure," on page 47

Chapter 4, "The DQSRVADM Procedure," on page 43

"DataFlux Integration Server Functions" on page 57

About Passwords for Integration Servers

If security has been implemented on your Integration Servers, you need to include user names and passwords in the procedures and function calls that access those servers. You can specify the passwords directly, in plain text, or you can specify encoded passwords. SAS recognizes encoded passwords and decodes them before it sends the passwords to the Integration Servers.

The following example shows how you can encode a password and use that password in a call to the DQSRVSVC procedure:

```
/*   Encode password in file.  */
filename pwfile 'c:\dataEntry01Pwfile';
proc pwencode in='Oe3s2m5' out=pwfile;
run;

/*  Load encoded password into macro variable.  */
data _null_;
   infile pwfile obs=1 length=l;
   input @;
   input @1 line $varying1024. l;
   call symput ('dbpass', substr(line,1,l));
run;

/*  Run service on secure Integration Server  */
```

```
proc dqsrvsvc
   service="cleanseCorpName"  host="entryServer1"
   userid="DataEntry1"              password="&dbpass"
   data=corpName                   out=corpNameClean;
run;
```

For further information on the PWENCODE procedure, including information on the format of the encoded passwords, see the *Base SAS Procedures Guide*.

Understand the Setup File

To access a Quality Knowledge Base, your SAS programs refer to path specifications in the setup file dqsetup.txt. The location of the setup file is specified by the system option DQSETUPLOC=. The value of the system option can be the path to dqsetup.txt, or a path to the root directory of your Quality Knowledge Base.

In the z/OS operating environment, the setup file DQSETUP is a member of a SAS Data Quality Configuration PDS.

Note: SAS programs that run jobs and services on DataFlux Integration Servers do not reference the setup file. Nor do they directly access a Quality Knowledge Base. △

If you move your Quality Knowledge Base, make sure that you change the value of DQSETUPLOC= accordingly.

If your site uses multiple Quality Knowledge Bases, make sure that your programs set DQSETUPLOC= to point to the intended setup file.

Edit the SAS Data Quality Server Setup File

The data quality setup file consists of a table with two columns. The first column lists file access methods. The second column provides fully-qualified paths to the contents of the Quality Knowledge Base.

If you edit the setup file, always be sure to do these tasks:

□ Include a semicolon at the end of each fully-qualified path.

□ Specify the DISK access method.

□ In the Windows and UNIX operating environments, do not change the last directory or filename in any path. For the z/OS operating environment, do not change the PDS names, and don’t change the names of their contents. For example, in the following entry in the setup file, you would retain the GRAMMAR PDS name and not change any member names inside that PDS:

```
DISK SAS91.DQ.GRAMMAR;
```

If you would like to add comment lines to your setup file, specify an exclamation point (!) as the first non-blank character in each line that contains comments. When an exclamation point is detected in this position, the software ignores any other text on that line.

You can insert blank lines into the setup file.

In the setup file, the maximum record (line) length is 1024 characters.

Load and Unload Locales

You need to load and unload locales in order to run data-cleansing programs in SAS. Conversely, you do not need to load locales if your SAS programs run jobs and services on Integration Servers from DataFlux (a SAS company).

Before you run data-cleansing programs in SAS, you load locales into memory using the AUTOCALL macro %DQLOAD (see "%DQLOAD AUTOCALL Macro" on page 51). The macro sets the value of the system options DQSETUPLOC= and DQLOCALE= and loads the specified locales into local memory. DQSETUPLOC= specifies the location of the setup file, and DQLOAD= specifies an ordered list of locales.

The order of locales in the locale list is pertinent only when one of the following conditions is true:

- □ A locale is not specified by name.
- □ The specified locale is not loaded into memory.
- □ Input data is insufficient for the DQLOCALEGUESS function (see"DQLOCALEGUESS Function" on page 64) to determine the best locale for that data.

If a locale cannot be established, SAS searches the list of locales and references the first definition it finds that has the specified name.

Note that you can change the values of the system options DQSETUPLOC= and DQLOCALE=, but doing so does not load different locales into memory. For this reason, it is recommended that you use %DQLOAD to change the values of the two data quality system options.

If you change locale files in the Quality Knowledge Base using the dfPower Customize software from DataFlux, make sure that you reload macros into memory with %DQLOAD before cleansing data.

After you submit your data-cleansing programs, you can restore the memory by unloading locales using the AUTOCALL macro %DQUNLOAD (see "%DQUNLOAD AUTOCALL Macro" on page 53).

New locales, and updates to existing locales, are provided periodically by DataFlux in the form of a new Quality Knowledge Base, which you can download from the following Web address:

```
www.dataflux.com/QKB
```

Specify Definitions

To specify definitions in your SAS data cleansing programs, you need to know the names of the definitions that are available in a particular locale. Use the AUTOCALL macro %DQPUTLOC (see "%DQPUTLOC AUTOCALL Macro" on page 52) to display information on a locale that is currently loaded into memory. The function DQLOCALEINFOGET (see "DQLOCALEINFOGET Function" on page 65) serves the same purpose.

Use the function DQLOCALEINFOLIST to display a list of definitions in a specified locale (see "DQLOCALEINFOLIST Function" on page 65).

If you are unsure of the locale that best fits your data, use the function DQLOCALEGUESS to return the name of the locale (see "DQLOCALEGUESS Function" on page 64).

Configure Your SAS Session for Data Quality

Use the DQOPTIONS= system option to configure your SAS session for data quality. You can specify the option in your SAS start-up command or in your SAS configuration file, SASV*9*.CFG. DQOPTIONS= enables you to specify one or more option-value pairs that pertain to the operation of the SAS Data Quality Server software.

The DQOPTIONS= system option currently enables two option-value pairs. To improve data transfer performance, the value DQSRVPROTOCOL=WIRELINE is required in the z/OS operating environment, for DataFlux Integration Servers version 8.1.1 or newer. In other operating environments, the default SOAP protocol is recommended.

TRANSCODE=IGNORE | WARN specifies that transcoding errors between character sets are to be ignored, and SAS processing is allowed to continue. By default, transcoding errors terminate SAS processing.

To learn more about DQOPTIONS=, see "DQOPTIONS=" on page 98.

Transform Data with Schemes

A scheme is a data set that you create to transform the values of a character variable. You first create a scheme data set, and then you apply it to the data. The transformation converts related groups of values into a single value.

The DQSCHEME procedure (see Chapter 3, "The DQSCHEME Procedure," on page 31) is used to create and apply schemes. You can create and apply multiple schemes in a single procedure step. The DQSCHEMEAPPLY function and CALL routine are also used to apply schemes (see "DQSCHEMEAPPLY CALL Routine" on page 77). You can also display, create, apply, and edit schemes in the dfPower Studio software from DataFlux (a SAS company).

Creating Schemes

Before you apply a scheme (see "Applying Schemes" on page 10), you first create the scheme data set. Schemes are created with the CREATE statement in the DQSCHEME procedure. When you submit a CREATE statement, PROC DQSCHEME creates match codes and assigns cluster numbers. A unique cluster number is assigned to each group of two or more input values that generate the same match code. After cluster numbers are assigned, the transformation value is determined. The transformation value is the most common value in each cluster.

Note: During scheme creation, the DQSCHEME procedure evaluates the definition of each input variable in each CREATE statement. An error message is generated if the defined length of an input variable exceeds 1024 bytes. △

Scheme data sets are created in SAS format or BFD format. BFD stands for Blue Fusion Data, which is a format that is recognized by SAS and by the dfPower Studio software. The SAS Data Quality Server software can create, apply, and display schemes in SAS format or BFD format. The dfPower Studio software can create, apply, display, and edit schemes in BFD format only. In the z/OS operating environment, the SAS Data Quality Server software can create, apply, and display schemes in SAS format; schemes in BFD format can be applied.

About Analysis Data Sets

Analysis data sets describe how a transformation would take place, without actually transforming the data; they enable you to experiment with different options to arrive at a scheme that provides optimal data cleansing. Analysis data sets are generated by specifying the ANALYSIS= option in the CREATE statement of the DQSCHEME procedure.

The key to optimizing a scheme is to choose the sensitivity value that best suits your data and your goal. You can create a series of analysis data sets using different sensitivity values to compare the results. Changing the sensitivity value changes the clustering of input values, as described in "About Sensitivity" on page 14.

When you decide on a sensitivity level you can create the scheme data set by replacing the ANALYSIS= option with the SCHEME= option in the CREATE statement. For further information on sensitivity values and how they affect the creation of match codes and clusters, see "About Sensitivity" on page 14.

The analysis data set contains one observation for each unique input value. Any adjacent blank spaces are removed from the input values. The COUNT variable describes the number of occurrences of that value. The CLUSTER variable designates a cluster number for each input character value.

The CLUSTER variable contains a value only when two or more *different* input values generate the same match code. An input value that is repeated receives a cluster number only if another value generates the same match code.

You can specify the INCLUDE_ALL option in the CREATE statement to include all input values in the scheme, including the unique input values that did not receive a cluster number in the analysis data set.

Applying Schemes

After you create a scheme data set (see "Creating Schemes" on page 9), you apply it to an input variable to transform its values. You can apply a scheme with the APPLY statement in the DQSCHEME procedure (see "APPLY Statement" on page 35), or with the DQSCHEMEAPPLY function or CALL routine (see "DQSCHEMEAPPLY Function" on page 81). Use the CALL routine rather than the function if you want to return the number of transformations that occurred during the application of the scheme.

The scheme data set consists of the DATA and STANDARD variables. The DATA variable contains the input character values that were used to create the scheme. The STANDARD variable contains the transformation values. All of the DATA values in a given cluster have the same STANDARD value. The STANDARD values are the values that were the most common values in each cluster when the scheme was created.

When you apply a scheme to a SAS data set, an input value is transformed when the it matches a DATA value in the scheme. The transformation replaces the input value with the transformation value.

The lookup method determines how the input value is matched to the DATA values in the scheme. The SCHEME_LOOKUP option or argument specifies that the match must be exact, case-insensitive, or consist of a match between the match codes of the input value and the match codes of the DATA values in the scheme. When a match occurs, any adjacent blank spaces in the transformation value are replaced with single blank spaces; then the value is written into the output data set. If no match is found for an input value, that exact value is written into the output data set.

You can specify the MODE argument or the MODE= option to apply schemes in one of two modes: *phrase* or *element*. Applying a scheme by phrase compares the entire input value (or the match code of the entire value) to the values (or match codes) in the scheme. Phrase is the default scheme apply mode.

When you apply a scheme by element, each element in the input value (or match code of each element) is compared to the values (or match codes) in the scheme. Applying schemes by element enables you to change one or more elements in an input value, without changing any of the other elements in that value.

The file format of a scheme is important when that scheme is applied. In the z/OS operating environment, schemes must be created and applied in SAS format. Schemes that are stored in a PDS in BFD format can be applied, and schemes in BFD format can be converted to SAS format using the CONVERT statement in the DQSCHEME procedure.

Note: Schemes in BFD format cannot be created or displayed in the z/OS operating environment. △

About Meta Options

Meta options are stored in the scheme when the scheme is created; they provide default values for certain options of the APPLY statement of the DQSCHEME procedure or default arguments for the DQSCHEMEAPPLY function or CALL routine. Default values are stored for the lookup mode (SCHEME_LOOKUP option or argument), apply mode (MODE option or argument), match definition, and sensitivity level. The values of the meta options are superseded when other values are specified in the APPLY statement or in the DQSCHEMEAPPLY function or CALL routine.

When the scheme is applied, the meta options for the match definition and sensitivity value are valid only when the scheme is applied with match-code lookup (when the value of SCHEME_LOOKUP isUSE_MATCHDEF).

The meta options are stored differently depending on the scheme format. For schemes in SAS format, the meta options are stored in the data set label. For schemes in BFD format, the meta options are stored within the scheme itself.

Note: In programs that create schemes in SAS format, do not specify a data set label; doing so deletes the meta options. △

The meta options are stored using the following syntax:

"*lookup-method*" "*apply-mode*" "*sensitivity-level*" "*match-definition*"

lookup-method

Valid values are as follows:

EM — specifies that the default value of the SCHEME_LOOKUP option or argument is EXACT. For an input value to be transformed, that value must exactly match a DATA value in the scheme.

IC — SCHEME_LOOKUP=IGNORE_CASE.

UM — SCHEME_LOOKUP=USE_MATCHDEF. Match codes are created and compared for all input values and all DATA values in the scheme.

apply-mode
Valid values are as follows:

E	specifies that the default value of the MODE option or argument is ELEMENT.
P	MODE=PHRASE.

sensitivity-level
specifies the amount of information in the match codes that is generated when SCHEME_LOOKUP=USE_MATCHDEF. Valid values range from 50 to 95.

match-definition
specifies the name of the default match definition that will be used when the value of the SCHEME_LOOKUP option or argument is USE_MATCHDEF.

For example, the following meta options string specifies that, by default, the scheme uses match-code lookup, applied by phrase, using the NAME match definition and a sensitivity level of 80:

```
"UM" "P" "80" "NAME"
```

Create Match Codes

Match codes are encoded representations of character values that are used for analysis, transformation, and standardization of data. Match codes are created by the following procedures and functions:

PROC DQMATCH
creates match codes for one or more variables or parsed tokens that have been extracted from a variable. The procedure can also assign cluster numbers to values with identical match codes. For syntax information, see "DQMATCH Procedure Syntax" on page 19.

DQMATCH
generates match codes for a variable. See "DQMATCH Function" on page 67.

DQMATCHPARSED
generates match codes for tokens that have been parsed from a variable. See "DQMATCHPARSED Function" on page 69.

Match codes are created by the DQMATCH procedure and by the functions DQMATCH and DQMATCHPARSED. The functions DQMATCH and DQMATCHPARSED return one match code for one input character variable. With these tools you can create match codes for an entire character value or a parsed token extracted from a character value

During processing, match codes are generated according to the specified locale, match definition, and sensitivity.

The locale identifies the language and geographical region of the source data. For example, the locale ENUSA specifies that the source data uses the English language as it is used in the United States of America.

The match definition in the Quality Knowledge Base identifies the category of the data and determines the content of the match codes. Examples of match definitions are named ADDRESS, ORGANIZATION, and DATE(YMD). To determine the match definitions that are available in a Quality Knowledge Base, consult the QKB documentation from DataFlux (a SAS company), or use the function DQLOCALEINFOLIST to return the names of the match definitions in your locale. Use the function if your site modifies the default Quality Knowledge Base using the dfPower Customize software from DataFlux.

The sensitivity level is a value between 0 and 99 that determines the amount of information that is captured in the match code, as described in “About Sensitivity” on page 14.

If two or more match codes are identical, a cluster number can be assigned to a specified variable, as described in “About Clusters” on page 14. The content of the output data set is determined by option values. You can choose to include values that generate unique match codes and you can choose to include and add a cluster number to blank or missing values. You can also concatenate multiple match codes.

Note also that match codes are also generated internally when you create a scheme with PROC DQSCHEME, as described in “Transform Data with Schemes” on page 9. Note that match codes are created internally by the DQSCHEME procedure, the DQSCHEMEAPPLY function, and the DQSCHEMEAPPLY CALL routine. These match codes are used in the process of creating or applying a scheme, as described in “Transform Data with Schemes” on page 9.

How Match Codes Are Created

You can create two types of match codes:

□ *Simple match codes* are created from a single input character variable.

□ *Composite match codes* consist of a concatenation of match codes from two or more input character variables. Then the separate match codes are concatenated into a composite match code. You have the option of specifying that a delimiter, in the form of an exclamation point (!), is to be inserted between the simple match codes that comprise the combined match code (via the DELIMITER option or argument).

To create simple match codes, you specify one CRITERIA statement, with one input variable identified in the VAR= option and one output variable identified with the MATCHCODE= option. Composite match codes are similar, except that you specify multiple CRITERIA statements for multiple variables, and all of those CRITERIA statements specify the same output variable in their respective MATCHCODE= options.

The SAS Data Quality Server software creates match codes using these general steps:

1. Parse the input character value to identify tokens.
2. Remove insignificant words.
3. Remove some of the vowels. Remove fewer vowels when a scheme-build match definition has been specified, as described in “About the Scheme Build Match Definitions” on page 16.
4. Standardize the format and capitalization of words.
5. Create the match code by extracting the appropriate amount of information from one or more tokens, based on the specified match definition and level of sensitivity.

Certain match definitions skip some of these steps.

Note: When you work with two or more data sets that you intend to analyze together or join using match codes, be sure to use identical sensitivities and match definitions when you create the match codes in each data set. △

About the Length of Match Codes

Match codes can vary in length between 1 and 1024 bytes. The length is determined by the specified match definition. If you receive a message in the SAS log that states that match codes have been truncated, you should extend the length of your match code variable. Truncated match codes will not produce accurate results.

About Clusters

Clusters are numbered groups of values that generate identical match codes or that have an exact match of characters. Clusters are used in the creation of schemes using the DQSCHEME procedure. The cluster with the greatest number of members becomes the transformation value for the scheme.

Householding with PROC DQMATCH

You can use the DQMATCH procedure to generate cluster numbers as it generates match codes. One important application for clustering in PROC DQMATCH is commonly referred to as householding, where members of a family or household are identified in clusters that are based on multiple criteria and conditions.

To establish the criteria and conditions for householding, use multiple CRITERIA statements and CONDITION= options within those statements. The integer values of the CONDITION= options are reused across multiple CRITERIA statements to establish groups of criteria. Within each group, match codes are created for each criteria. If a source row is to receive a cluster number, all of the match codes in the group must match all of the codes in another source row. The match codes within a group are therefore evaluated with a logical AND.

If more than one condition number is specified across multiple CRITERIA statements, then you have multiple groups, and multiple groups of match codes. In this case, source rows receive cluster numbers when any of its groups matches any other group in another source row. The groups are therefore evaluated with a logical OR.

For an example of householding, assume that a data set that contains customer information. To assign cluster numbers you use two groups of two CRITERIA statements. One group (condition 1) uses two CRITERIA statements to generate match codes based on the names of individuals and an address. The other group (condition 2) generates match codes based on organization name and address. A cluster number is assigned to a source row when either pair of match codes matches at least one group matches the match codes from another source row. The code and output for this example is provided in Example 5 on page 28.

Clustering with Exact Criteria

The EXACT option of the CRITERIA statement in PROC DQMATCH enables you to use exact character matches as part of your clustering criteria. Exact character matches are helpful in situations where you want to assign cluster numbers using a logical AND of an exact number and the match codes of a character variable. For example, you could assign cluster numbers using two criteria, one using an exact match on a customer ID values, the other using a match code generated from customer names. Information on the syntax of the EXACT option is provided in “CRITERIA Statement” on page 21.

About Sensitivity

The amount of information contained in match codes is determined by a specified sensitivity level. Changing the sensitivity level allows you to change what is considered a match. Match codes that are created at lower levels of sensitivity capture little information about the input values. The result is more matches, fewer clusters (see “About Clusters” on page 14), and more values in each cluster. At higher sensitivity levels, input values must be more similar to receive the same match code. Clusters are more numerous, and the number of entries in each cluster is smaller.

In some data cleansing jobs, a lower sensitivity value is needed. For example, if you wanted to transform the following names to a single consistent value using a scheme, you would need to specify a lower sensitivity level:

```
Patricia J. Fielding
Patty Fielding
Patricia Feelding
Patty Fielding
```

In this, all four values would be assigned to the same cluster and would be transformed to the most-common value, **`Patty Fielding`**.

In other cases, a higher sensitivity level is needed. For example, if you were collecting customer data based on account numbers, you would want to cluster on individual account numbers. A high sensitivity value would be needed.

In the SAS Data Quality Server software, sensitivity values range from 50 to 95, and the default value is 85.

To arrive at the sensitivity level that fits your data and your application, run tests with DQMATCH or create analysis data sets with PROC DQSCHEME.

About Locale Definitions

Using Parse Definitions

Parse definitions are referenced when you want to create parsed input values. Parsed input values are delimited so that the elements in those values can be associated with named tokens. After parsing, specific contents of the input values can be returned by specifying the names of tokens.

Parse definitions and tokens are referenced by the following functions:

- □ “DQPARSE Function” on page 71.
- □ “DQPARSEINFOGET Function” on page 72.
- □ “DQTOKEN Function” on page 94.
- □ “DQPARSETOKENGET Function” on page 73.
- □ “DQPARSETOKENPUT Function” on page 74

For a brief example of how tokens are assigned and used, see “Specify Definitions” on page 8.

Parsing a character value assigns tokens only when the content in the input value meets the criteria in the parse definition. Parsed character values can therefore contain empty tokens. For example, three tokens are empty when you use the DQPARSE function to parse the character value **`Ian M. Banks`**, using the NAME parse definition in the ENUSA locale. The resulting token/value pairs are as follows:

NAME PREFIX	empty
GIVEN NAME	Ian
MIDDLE NAME	M.

FAMILY NAME	Banks
NAME SUFFIX	empty
NAME APPENDAGE	empty

Note: For parse definitions that work with dates, such as DATE (DMY) in the ENUSA locale, input values must be character data rather than SAS dates. △

About the Global Parse Definitions

Global parse definitions contain a standard set of parse tokens that enable the analysis of similar data from different locales. For example, the ENUSA locale and the DEDEU locale both contain the parse definition ADDRESS (GLOBAL). The parse tokens are the same in both locales. This global parse definition enables the combination of parsed character data from multiple locales.

All global parse definitions are identified by the (GLOBAL) suffix.

Using Match Definitions

Match definitions are referenced during the creation of match codes. Match codes provide a variable method of clustering similar input values as a basis for data cleansing jobs such as the application of schemes.

When you create match codes, you determine the number of clusters (values with the same match code) and the number of members in each cluster by specifying a sensitivity level. The default sensitivity level is specified by the procedure or function, rather than the match definition. For information on sensitivity levels, see “About Sensitivity” on page 14.

Match definitions are referenced by the following procedures and functions:

□ Chapter 2, “The DQMATCH Procedure,” on page 19.

□ Chapter 3, “The DQSCHEME Procedure,” on page 31.

□ “DQMATCH Function” on page 67.

□ “DQMATCHINFOGET Function” on page 68.

□ “DQMATCHPARSED Function” on page 69.

When you create match codes for parsed character values, your choice of match definition depends on the parse definition that was used to parse the input character value. To determine the parse definition that is associated with a given match definition, use the “DQMATCHINFOGET Function” on page 68.

Note: For match definitions that work with dates, such as DATE (MDY) in the ENUSA locale, input values must be character data rather than SAS dates. △

About the Scheme Build Match Definitions

Locales contain certain match definitions that are recommended for use in the DQSCHEME procedure because they produce more desirable schemes. The names of these scheme-build match definitions always end with “(SCHEME BUILD)”.

Scheme-build match definitions are advantageous because they create match codes that contain more vowels. Match codes that contain more vowels result in more clusters with fewer members in each cluster, which in turn results in a larger, more specific set of transformation values.

When you are using the DQMATCH procedure or function to create simple clusters, it is better to have fewer vowels in the match code. For example, when using the CITY match definition in PROC DQMATCH, the values Baltimore and Boltimore receive the same match codes. The match codes would differ if you used the match definition CITY (SCHEME BUILD).

Using Case and Standardization Definitions

Case and standardization definitions are applied to character values to make them more consistent for the purposes of display or in preparation for transforming those values with a scheme.

Case definitions are referenced by the “DQCASE Function” on page 59. Standardization definitions are referenced by the “DQSTANDARDIZE Function” on page 92.

Case definitions transform the capitalization of character values. For example, the case definition Proper in the ENUSA locale takes as input any general text, capitalizes the first letter of each word, and uses lowercase for the other letters in the word, while recognizing and retaining or transforming various words and abbreviations into uppercase. Other case definitions, such as PROPER – ADDRESS, apply to specific text content.

Standardization definitions standardize the appearance of specific data values. In general, words are capitalized appropriately based on the content of the input character values. Also, adjacent blank spaces are removed, along with unnecessary punctuation. Additional standardizations might be made for specific content. For example, the standardization definition STATE (FULL NAME) in the locale ENUSA converts abbreviated state names to full names in uppercase.

About the Standardization of Dates in the EN Locale

In the EN locale, dates are standardized to two-digit days (00–31), two-digit months (01–12), and four-digit years. Input dates must be character values rather than SAS dates. Spaces separate (delimit) the days, months, and years, as shown in the following table.

Table 1.1 Examples of Date Standardizations

Input Date	Standardization Definition	Standardized Date
July04, 03	Date (MDY)	07 04 2003
July 04 04	Date (MDY)	07 04 1904
July0401	Date (MDY)	07 04 2001
04.07.02	Date (DMY)	04 07 2002
04-07-2004	Date (DMY)	04 07 2004
03/07/04	Date (YMD)	2003 07 04

Two-digit year values are standardized as follows. If an input year is greater than 00 and less than or equal to 03, the standardized year will be 2000, 2001, 2002, or 2003. Two-digit input year values that are greater than or equal to 04 and less than or equal to 99 will be standardized into the range of 1904–1999. For example, an input year of 03 is standardized as 2003. An input year of 04 is standardized as 1904. These standardizations are not affected by the value of the SAS system option YEARCUTOFF=.

Using Gender Analysis, Locale Guess, and Identification Definitions

Gender analysis, locale guess, and identification definitions enable you make determinations about character values. With these definitions you can determine:

- □ The gender of an individual based on a name value.
- □ The locale that is the most suitable for a given character value.
- □ The category of a value, which is chosen from a set of available categories.

Gender analysis definitions determine the gender of an individual based on that individual's name. The gender is determined to be unknown if the first name is used by both males and females, if no other clues are provided in the name or if conflicting clues are found. Gender analysis definitions are referenced by the "DQGENDER Function" on page 60.

Locale guess definitions allow the software to determine the locale that is most likely represented by a character value. All locales that are loaded into memory as part of the locale list are considered, but only if they contain the specified guess definition. If a definite locale determination cannot be made, the chosen locale is the first locale in the locale list. Locale guess definitions are referenced by the "DQLOCALEGUESS Function" on page 64.

Identification definitions are used to categorize character values. For example, using the Entity identification definition in the ENUSA locale, a name value can be determined to apply to an individual or an organization. Identification definitions are referenced by the "DQIDENTIFY Function" on page 63.

Using Pattern Analysis Definitions

Pattern analysis definitions enable you to determine whether an input character value contains characters that are alphabetic, numeric, non-alphanumeric (punctuation marks or symbols), or a mixture of alphanumeric and non-alphanumeric. The ENUSA locale contains two pattern analysis definitions. The pattern analysis definition WORD is referenced by the DQPATTERN function (see "DQPATTERN Function" on page 76) to generate one character of analytical information for each word in the input character value. The CHARACTER definition generates one character of analytical information for each character in the input character value.