SAS/ACCESS® 9.1.3 Supplement for HP Neoview

SAS/ACCESS for Relational Databases

SAS® Documentation

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2007. *SAS/ACCESS® 9.1.3 Supplement for HP Neoview (SAS/ACCESS for Relational Databases)*. Cary, NC: SAS Institute Inc.

SAS/ACCESS® 9.1.3 Supplement for HP Neoview (SAS/ACCESS for Relational Databases)

ISBN 978-1-59994-595-8

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, December 2007

SAS® Publishing provides a complete selection of books and electronic products to help customers use SAS software to its fullest potential. For more information about our e-books, e-learning products, CDs, and hard-copy books, visit the SAS Publishing Web site at **support.sas.com/pubs** or call 1-800-727-3228.

Contents

CHAPTER

1

SAS/ACCESS for HP Neoview

Introduction to SAS/ACCESS Interface to HP Neoview

This document describes *only* SAS/ACCESS Interface to HP Neoview. Use it as a supplement to the generic SAS/ACCESS documentation, *SAS/ACCESS for Relational Databases: Reference*.

LIBNAME Statement Specifics for HP Neoview

Overview

This section describes the LIBNAME statement that SAS/ACCESS Interface to HP Neoview supports. For a complete description of this feature, see the LIBNAME statement section in *SAS/ACCESS for Relational Databases: Reference*. Here is the LIBNAME statement syntax for accessing HP Neoview.

LIBNAME *libref* **neoview** <*connection-options*> <*LIBNAME-options*>;

Arguments

libref
is any SAS name that serves as an alias to associate SAS with a database, schema, server, or group of tables and views.

neoview
is the SAS/ACCESS engine name for SAS/ACCESS Interface to HP Neoview.

connection-options
provide connection information and control how SAS manages the timing and concurrence of the connection to the DBMS. When you use the LIBNAME statement on UNIX or Microsoft Windows, you can connect to HP Neoview Database Connectivity Service (NDCS) by connecting a client to a data source. Use *only one* of the following methods for each connection because they are mutually exclusive.

- specify SERVER=, SCHEMA=, PORT=, USER=, and PASSWORD=, or
- specify DSN=, USER=, and PORT=

Here is how these connection options are defined.

SERVER=<'>*server-name*<'>
specifies the server name or IP address of the HP Neoview server to which you want to connect. This server accesses the database that contains the tables and views that you want to access. If the server name contains spaces or nonalphanumeric characters, you must enclose it in quotation marks.

SCHEMA=<'>*schema-name*<'>
specifies the name of the schema. When you use it with SERVER= or PORT=, it is passed directly as a connection option to the database. When you use it with DSN=, it qualifies SQL statements as a LIBNAME= option. You can also use it as a data set option.

PORT=*port*
specifies the port number that is used to connect to the specified HP Neoview server. If you do not specify a port, the default of 18650 is used.

USER=<'>*Neoview-user-name*<'>
specifies the HP Neoview user name (also called the user ID) that you use to connect to your database. If the user name contains spaces or nonalphanumeric characters, you must enclose the user name in quotation marks.

PASSWORD=<'>*Neoview-password*<'>
specifies the password that is associated with your HP Neoview user name. If the password contains spaces or nonalphanumeric characters, you must enclose the password in quotation marks. You can also specify PASSWORD= with the PWD=, PASS=, and PW= aliases.

DSN=<'>*Neoview-data-source*<'>
specifies the configured HP Neoview ODBC datasource to which you want to connect. Use this option if you have existing HP Neoview ODBC datasources that are configured on your client. This method requires additional setup—either through the ODBC Administrator control panel on Windows platforms or through the MXODSN file on UNIX platforms. It is recommended that you use this connection method only if you have existing, functioning datasources that have been defined.

LIBNAME-options
define how SAS processes DBMS objects. Some LIBNAME options can enhance performance; others determine locking or naming behavior. The following table describes the LIBNAME options that are supported for HP Neoview and presents default values where applicable. See the section about the SAS/ACCESS LIBNAME statement in *SAS/ACCESS for Relational Databases: Reference* for detailed information about these options.

Table 1.1 SAS/ACCESS LIBNAME Options for HP Neoview

Option	Default Value
ACCESS=	none
AUTOCOMMIT=	operation-specific
CONNECTION=	UNIQUE
CONNECTION_GROUP=	none
CONNECTION_TIMEOUT=	0
DBCOMMIT=	1000 (inserting) or 0 (updating)
DBCONINIT=	none
DBCONTERM=	none
DB_CREATE_TABLE_OPTS=	none
DBGEN_NAME=	DBMS
DBINDEX=	YES
DBLIBINIT=	none
DBLIBTERM=	none
DBMAX_TEXT=	1024
DBNULLKEYS=	YES
DBPROMPT=	NO
DEFER=	NO
DELETE_MULT_ROWS=	NO
DIRECT_EXE=	none
DIRECT_SQL=	YES

Option	Default Value
IGNORE_ READ_ONLY_COLUMNS=	NO
INSERTBUFF=	automatically calculated based on row length
MULTI_DATASRC_OPT=	NONE
PRESERVE_COL_NAMES=	see Naming Conventions for HP Neoview“Naming Conventions for HP Neoview” on page 10
PRESERVE_TAB_NAMES=	see Naming Conventions for HP Neoview“Naming Conventions for HP Neoview” on page 10
QUERY_TIMEOUT=	0
QUOTE_CHAR=	none
READBUFF=	automatically calculated based on row length
REREAD_EXPOSURE=	NO
SCHEMA=	none
SPOOL=	YES
SQL_FUNCTIONS=	none
STRINGDATES=	NO
TRACE=	NO
TRACEFILE=	none
UPDATE_MULT_ROWS=	NO
UTILCONN_TRANSIENT=	NO

HP Neoview LIBNAME Statement Examples

In this example, SERVER=, SCHEMA=, USER=, and PASSWORD= are connection options.

```
libname mydblib neoview server=ndcs1 schema=USR user=neo1 password=neopwd1;
```

In this next example, DSN=, USER=, and PASSWORD= are connection options.

```
libname mydblib neoview DSN=TDM_Default_DataSource user=neo1 password=neopwd1;
```

Data Set Options for HP Neoview

The following table describes the data set options that are supported for HP Neoview and provides default values where applicable. See the section about data set options in *SAS/ACCESS for Relational Databases: Reference* for detailed information about these options.

Table 1.2 SAS/ACCESS Data Set Options

Option	Default Value
DBCOMMIT=	LIBNAME option setting
DBCONDITION=	none

Option	Default Value
DBCREATE_TABLE_OPTS=	LIBNAME option setting
DBFORCE=	NO
DBGEN_NAME=	DBMS
DBINDEX=	LIBNAME option setting
DBKEY=	none
DBLABEL=	NO
DBMASTER=	none
DBMAX_TEXT=	1024
DBNULL=	YES
DBNULLKEYS=	LIBNAME option setting
DBPROMPT=	LIBNAME option setting
DBSASLABEL=	COMPAT
DBSASTYPE=	see Data Types for HP Neoview“Data Types for HP Neoview” on page 11
DBTYPE=	see Data Types for HP Neoview“Data Types for HP Neoview” on page 11
ERRLIMIT=	1
IGNORE_ READ_ONLY_COLUMNS=	NO
INSERTBUFF=	LIBNAME option setting
NULLCHAR=	SAS
NULLCHARVAL=	a blank character
PRESERVE_COL_NAMES=	LIBNAME option setting
QUALIFIER=	LIBNAME option setting
QUERY_TIMEOUT=	LIBNAME option setting
READBUFF=	LIBNAME option setting
SASDATEFMT=	none
SCHEMA=	LIBNAME option setting

Pass-Through Facility Specifics for HP Neoview

Overview

See the Pass-Through Facility section in *SAS/ACCESS for Relational Databases: Reference* for general information about this feature.

Here are the Pass-Through Facility specifics for the HP Neoview interface.

- **`NEOVIEW`** is the *dbms-name*.
- The CONNECT statement is required.

- PROC SQL supports multiple connections to HP Neoview. If you use multiple simultaneous connections, you must use the *alias* argument to identify the different connections. If you do not specify an alias, the default **neoview** alias is used.
- The CONNECT statement *database-connection-arguments* are identical to its LIBNAME connection-options.
- You can use the SCHEMA= option only with the SERVER= and PORT= connection options. It is not valid with DSN= in a pass-through connection.

CONNECT Statement Example

This example, uses the DBCON alias to connection to the **mynpssrv** HP Neoview server and execute a query. The connection alias is optional.

```
proc sql;
   connect to neoview as dbcon
   (server=ndcs1 schema=TEST user=neo1 password=neopwd1);
select * from connection to dbcon
   (select * from customers where customer like '1%');
quit;
```

Special Catalog Queries

SAS/ACCESS Interface to HP Neoview supports the following special queries, which you can use to call the ODBC-style catalog function APIs. Here is the general format of the special queries.

Neoview::SQLAPI "*parameter 1*","*parameter n*"

Neoview::
is required to distinguish special queries from regular queries.

SQLAPI
is the specific API that is being called. Neither HP Neoview:: nor SQLAPI are case sensitive.

"*parameter n*"
is a quoted string that is delimited by commas.

Within the quoted string, two characters are universally recognized: the percent sign (%) and the underscore (_). The percent sign matches any sequence of zero or more characters, and the underscore represents any single character. To use either character as a literal value, you can use the backslash character (\) to escape the match characters. For example, the following call to SQLTables usually matches table names such as myatest and my_test:

```
select * from connection to neoview (NEOVIEW::SQLTables "test","","my_test");
```

Use the escape character to search only for the table, my_test:

```
select * from connection to neoview (NEOVIEW::SQLTables "test","","my\_test");
```

SAS/ACCESS Interface to HP Neoview supports these special queries:

Neoview::SQLTables <"*Catalog*", "*Schema*", "*Table-name*", "*Type*">

returns a list of all tables that match the specified arguments. If you do not specify any arguments, all accessible table names and information are returned.

Neoview::SQLColumns <"*Catalog*", "*Schema*", "*Table-name*", "*Column-name*">
returns a list of all columns that match the specified arguments. If you do not specify any argument, all accessible column names and information are returned.

Neoview::SQLPrimaryKeys <"*Catalog*", "*Schema*", "*Table-name*">
returns a list of all columns that compose the primary key that matches the specified table. A primary key can be composed of one or more columns. If you do not specify a table name, this special query fails.

Neoview::SQLSpecialColumns <"*Identifier-type*", "*Catalog-name*", "*Schema-name*", "*Table-name*", "*Scope*", "*Nullable*">
returns a list of the optimal set of columns that uniquely identify a row in the specified table.

Neoview::SQLStatistics <"*Catalog*", "*Schema*", "*Table-name*">
returns a list of the statistics for the specified table name, with options of SQL_INDEX_ALL and SQL_ENSURE set in the SQLStatistics API call. If you do not specify any table name argument, this special query fails.

Neoview::SQLGetTypeInfo
returns information about the data types that the HP Neoview server supports.

Passing SAS Functions to HP Neoview

SAS/ACCESS Interface to HP Neoview passes the following SAS functions to the data source for processing. Where the HP Neoview function name differs from the SAS function name, the HP Neoview name appears in parentheses. See the section about optimizing SQL usage in *SAS/ACCESS for Relational Databases: Reference* for information.

ABS
ARCOS (ACOS)
ARSIN (ASIN)
ATAN
ATAN2
AVG
BYTE (CHAR)
CEIL(CEILING)
COALESCE
COMPRESS (REPLACE)
COS
COSH
COUNT
DAY
EXP
FLOOR
HOUR
INDEX (LOCATE)
LEFT (LTRIM)

LOG
LOG10
LOWCASE (LOWER)
MAX
MIN
MINUTE
MOD
MONTH
REPEAT
QTR
SECOND
SIGN
SIN
SINH
SQRT
STRIP (TRIM)
SUBSTR
SUM
TAN
TANH
TRANWRD (REPLACE)
TRIMN (RTRIM)
UPCASE (UPPER)
WEEKDAY (DAYOFWEEK)
YEAR

If SQL_FUNCTIONS=ALL, SAS/ACCESS Interface to HP Neoview can also pass these functions to HP Neoview:

DATE (CURRENT_DATE)
DATEPART (CAST)
DATETIME (CURRENT_TIMESTAMP)
LENGTH
ROUND
TIME (CURRENT_TIME)
TIMEPART (CAST)
TODAY (CURRENT_DATE)

Passing Joins to HP Neoview

For a multiple libref join to pass to HP Neoview, all of these components of the LIBNAME statements must match exactly:

user ID (USER=)
password (PASSWORD=)
server (SERVER=)
database (DATABASE=)

port (PORT=)

datasource (DSN=, if specified)

catalog (QUALIFIER=, if specified)

SQL functions (SQL_FUNCTIONS=)

See the section about performance considerations in *SAS/ACCESS for Relational Databases: Reference* for more information about when and how SAS/ACCESS Interface to HP Neoview passes joins to the DBMS.

Temporary Table Support for HP Neoview

General Information

See the section on the temporary table support in *SAS/ACCESS for Relational Databases: Reference* for general information about this feature.

Establishing a Temporary Table

To make full use of temporary tables, the CONNECTION=GLOBAL connection option is necessary. This option lets you use a single connection across SAS DATA steps and SAS procedure boundaries. This connection can also be shared between LIBNAME statements and the Pass-Through Facility. Because a temporary table exists only within a single connection, you need to be able to share this single connection among all steps that reference the temporary table. The temporary table cannot be referenced from any other connection.

You can currently use only a PROC SQL statement to create a temporary table. To use both the Pass-Through Facility and librefs to reference a temporary table, you must specify a LIBNAME statement before the PROC SQL step so that global connection persists across SAS steps and even across multiple PROC SQL steps. Here is an example:

```
proc sql;
   connect to neoview (dsn=NDCS1_DataSource
       user=myuser password=mypwd connection=global);
   execute (create volatile table temptab1 as select * from permtable ) by neoview;
quit;
```

At this point, you can refer to the temporary table by using either the Temp libref or the CONNECTION=GLOBAL option with a PROC SQL step.

Terminating a Temporary Table

You can drop a temporary table at any time or allow it to be implicitly dropped when the connection is terminated. Temporary tables do not persist beyond the scope of a single connection.

Examples

The following assumptions apply to the examples in this section:

- The DeptInfo table already exists on the DBMS that contains all your department information.
- One SAS data set contains join criteria that you want to use to extract specific rows from the DeptInfo table.
- The other SAS data set contains updates to the DeptInfo table.

These examples use the following librefs and temporary tables.

```
libname saslib base 'SAS-Data-Library';
libname dept neoview dsn=Users_DataSource user=myuser pwd=mypwd connection=global;

proc sql;
   connect to neoview (dsn=Users_DataSource user=myuser pwd=mypwd connection=global);
   execute (create volatile table temptab1 (dname char(20), deptno int)) by neoview;
quit;
```

This first example shows how to use a heterogeneous join with a temporary table to perform a homogeneous join on the DBMS instead of reading the DBMS table into SAS to perform the join. By using the table that was created previously, you can copy SAS data into the temporary table to perform the join.

```
proc sql;
   connect to neoview (dsn=Users_DataSource user=myuser pwd=mypwd connection=global);
   insert into dept.temptab1 select * from saslib.joindata;
   select * from dept.deptinfo info, dept.temptab1 tab
        where info.deptno = tab.deptno;
   /* remove the rows for the next example */
   execute (delete from temptab1) by neoview;
   quit;
```

In this next example, transaction processing on the DBMS occurs by using a temporary table instead of using either DBKEY= or MULTI_DATASRC_OPT=IN_CLAUSE with a SAS data set as the transaction table.

```
proc sql;
connect to neoview (dsn=Users_DataSource user=myuser pwd=mypwd connection=global);
   insert into dept.temptab1 select * from saslib.transdat;
   execute (update deptinfo d set dname = (select dname from temptab1)
            where d.deptno = (select deptno from temptab1)) by neoview;
quit;
```

Naming Conventions for HP Neoview

Since SAS 7, most SAS names can be up to 32 characters long. SAS/ACCESS Interface to HP Neoview supports table names and column names that contain up to 32 characters. If DBMS column names are longer than 32 characters, they are truncated to 32 characters. If truncating a column name would result in identical names, SAS generates a unique name by replacing the last character with a number. DBMS table names must be 32 characters or less because SAS does not truncate a longer name. If you already have a table name that is greater than 32 characters, it is recommended that you create a table view.

The PRESERVE_COL_NAMES= and PRESERVE_TAB_NAMES= options determine how SAS/ACCESS Interface to HP Neoview handles case sensitivity. By default, HP Neoview is not case sensitive, and all names default to uppercase.

HP Neoview objects include tables, views, and columns. Follow these naming conventions:

- A name must be from 1 to 128 characters long.
- A name must begin with a letter (A through Z, or a through z). However, if the name appears within double quotation marks, it can start with any character.
- A name cannot begin with an underscore (_). Leading underscores are reserved for system objects.
- Names are not case sensitive. For example, **`CUSTOMER`** and **`Customer`** are the same, but object names are converted to uppercase when they are stored in the HP Neoview database. If you enclose a name in quotation marks, it is case sensitive.
- A name cannot be an HP Neoview-reserved word, such as WHERE or VIEW.
- A name cannot be the same as another HP Neoview object that has the same type.

For more information, see your *HP Neoview SQL Reference Manual*.

Data Types for HP Neoview

Overview

Every column in a table has a name and a data type. The data type tells HP Neoview how much physical storage to set aside for the column and the form in which the data is stored.

This information includes information about HP Neoview data types, null and default values, and data conversions. For more information about HP Neoview data types and to determine which data types are available for your version of HP Neoview, see your *HP Neoview SQL Reference Manual*.

Note: SAS/ACCESS Interface to HP Neoview does not directly support HP Neoview INTERVAL types. Any columns using these types are read into SAS as character strings. △

String Data

CHAR(*n*)
: specifies a fixed-length column for character string data. The maximum length is 32,708 characters.

VARCHAR(*n*)
: specifies a varying-length column for character string data. The maximum length is 32,708 characters.

Numeric Data

LARGEINT
specifies a large integer. Values in a column of this type can range from –9223372036854775808 to +9223372036854775807.

SMALLINT
specifies a small integer. Values in a column of this type can range from –32768 through +32767.

INTEGER
specifies a large integer. Values in a column of this type can range from –2147483648 through +2147483647.

DOUBLE
specifies a floating-point number that is 64 bits long. Values in a column of this type can range from –1.79769E+308 to –2.225E-307 or +2.225E-307 to +1.79769E+308, or the value can be 0. This data type is stored the same way that SAS stores its numeric data type. Therefore, numeric columns of this type require the least processing when SAS accesses them.

FLOAT
specifies an approximate numeric column. The column stores floating-point numbers and designates from 1 through 52 bits of precision. Values in a column of this type can range from +/-2.2250738585072014e-308 to +/-1.7976931348623157e+308 stored in 8 bytes.

REAL
specifies a floating-point number that is 32 bits long. Values in a column of this type can range from approximately –3.4E38 to –1.17E-38 and +1.17E-38 to +3.4E38.

DECIMAL | DEC | NUMERIC
specifies a fixed-point decimal number. The precision and scale of the number determines the position of the decimal point. The numbers to the right of the decimal point are the scale, which cannot be negative or greater than the precision. The maximum precision is 38 digits.

Dates, Times, and Timestamps

SQL date and time data types are collectively called *datetime* values. The SQL data types for dates, times, and timestamps are listed here. Be aware that columns of these data types can contain data values that are out of range for SAS.

DATE
specifies date values. The range is 01-01-0001 to 12-31-9999. The default format is *YYYY-MM-DD*—for example, 1961–06–13. HP Neoview supports other formats for entering date data. For more information, see your *HP Neoview SQL Reference Manual*.

TIME
specifies time values in hours, minutes, and seconds to six decimal positions: hh:mm:ss[.nnnnnn]. The range is 00:00:00.000000 to 23:59:59.999999. However, due to the ODBC-style interface that SAS/ACCESS Interface to HP Neoview uses to communicate with the HP Neoview server, any fractional seconds are lost in the transfer of data from server to client.

TIMESTAMP
combines a date and time in the default format of *yyyy-mm-dd hh:mm:ss[.nnnnnn]*. For example, a timestamp for precisely 2:25 p.m. on January 25, 1991, would be 1991-01-25-14.25.00.000000. Values in a column of this type have the same ranges as those described for DATE and TIME.

HP Neoview Null Values

HP Neoview has a special value called NULL. An HP Neoview NULL value means an absence of information and is analogous to a SAS missing value. When SAS/ACCESS reads an HP Neoview NULL value, it interprets it as a SAS missing value.

You can define a column in an HP Neoview table so that it requires data. To do this in SQL, specify a column as NOT NULL. This action tells SQL to allow only a row to be added to a table if a value exists for the field. For example, NOT NULL assigned to the CUSTOMER field in the SASDEMO.CUSTOMER table does not allow a row to be added unless there is a value for CUSTOMER. When creating an HP Neoview table with SAS/ACCESS, you can use the DBNULL= data set option to indicate whether NULL is a valid value for specified columns.

For more information about how SAS handles NULL values, see "Potential Result Set Differences When Processing Null Data" in *SAS/ACCESS for Relational Databases: Reference*.

To control how SAS missing character values are handled by the DBMS, use the NULLCHAR= and NULLCHARVAL= data set options.

LIBNAME Statement Data Conversions

The following table shows the default SAS variable formats that SAS/ACCESS assigns to HP Neoview data types during input operations when you use the LIBNAME statement.

Table 1.3 LIBNAME Statement: Default SAS Formats for HP Neoview Data Types

HP Neoview Data Type	SAS Data Type	Default SAS Format
CHAR(*n*)	character	$*n*.
VARCHAR(*n*)	character	$*n*.
LONGVARCHAR(*n*)	character	$*n*.
DECIMAL(*p,s*)	numeric	*m.n*
NUMERIC(*p,s*)	numeric	*p,s*
SMALLINT	numeric	6.
INTEGER	numeric	11.
REAL	numeric	none
FLOAT(*p*)	numeric	*p*
DOUBLE PRECISION	numeric	none
LARGEINT	numeric	20.
DATE	numeric	DATE9.

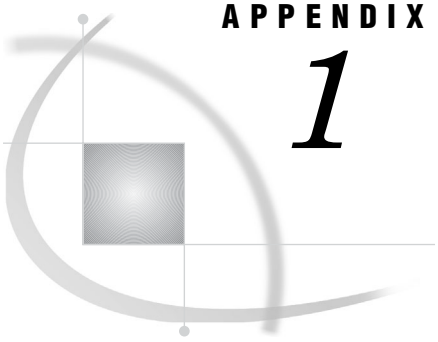
HP Neoview Data Type	SAS Data Type	Default SAS Format
TIME	numeric	TIME8.
TIMESTAMP	numeric	DATETIME*25.6*

The following table shows the default HP Neoview data types that SAS/ACCESS assigns to SAS variable formats during output operations when you use the LIBNAME statement.

Table 1.4 LIBNAME Statement: Default HP Neoview Data Types for SAS Variable Formats

SAS Variable Format	HP Neoview Data Type
m.n	DECIMAL (*m,n*)
other numerics	DOUBLE
$*n.*	VARCHAR(*n*)
datetime formats	TIMESTAMP
date formats	DATE
time formats	TIME

* *n* in HP Neoview data types is equivalent to *w* in SAS formats.



APPENDIX

1

Recommended Reading

Recommended Reading

Here is the recommended reading list for this title:

- *SAS/ACCESS Supplement for DB2 under z/OS*
- *SAS/ACCESS Supplement for DB2 under UNIX and PC Hosts*
- *SAS/ACCESS Supplement for Informix*
- *SAS/ACCESS Supplement for Microsoft SQL Server*
- *SAS/ACCESS Supplement for ODBC*
- *SAS/ACCESS Supplement for OLE DB*
- *SAS/ACCESS Supplement for Oracle*
- *SAS/ACCESS Supplement for Sybase*
- *SAS/ACCESS Supplement for Teradata*
- *SAS Language Reference: Concepts*
- *SAS Language Reference: Dictionary*
- *Base SAS Procedures Guide*
- SAS Companion that is specific to your operating environment

For a complete list of SAS publications, go to **support.sas.com/bookstore**. If you have questions about which titles you need, please contact a SAS Publishing Sales Representative at:

SAS Publishing Sales
SAS Campus Drive
Cary, NC 27513
Telephone: 1-800-727-3228
Fax: 1-919-531-9439
E-mail: **sasbook@sas.com**
Web address: **support.sas.com/bookstore**

Customers outside the United States and Canada, please contact your local SAS office for assistance.

Glossary

This glossary defines SAS software terms that are used in this document as well as terms that relate specifically to SAS/ACCESS software.

access descriptor
a SAS/ACCESS file that describes data that is managed by a data management system. After creating an access descriptor, you can use it as the basis for creating one or more view descriptors. See also view and view descriptor.

browsing data
the process of viewing the contents of a file. Depending on how the file is accessed, you can view SAS data either one observation (row) at a time or as a group in a tabular format. You cannot update data that you are browsing.

bulk load
to load large amounts of data into a database object, using methods that are specific to a particular DBMS. Bulk loading enables you to rapidly and efficiently add multiple rows of data to a table as a single unit.

client
(1) a computer or application that requests services, data, or other resources from a server. (2) in the X Window System, an application program that interacts with the X server and can perform tasks such as terminal emulation or window management. For example, SAS is a client because it requests windows to be created, results to be displayed, and so on.

column
in relational databases, a vertical component of a table. Each column has a unique name, contains data of a specific type, and has certain attributes. A column is analogous to a variable in SAS terminology.

column function
an operation that is performed for each value in the column that is named as an argument of the function. For example, AVG(SALARY) is a column function.

commit
the process that ends a transaction and makes permanent any changes to the database that the user made during the transaction. When the commit process occurs, locks on the database are released so that other applications can access the changed data. The SQL COMMIT statement initiates the commit process.

DATA step view
a type of SAS data set that consists of a stored DATA step program. Like other SAS data views, a DATA step view contains a definition of data that is stored elsewhere; the view does not contain the physical data. The view's input data can come from one or more sources, including external files and other SAS data sets. Because a DATA step view only reads (opens for input) other files, you cannot update the view's underlying data.

data type
a unit of character or numeric information in a SAS data set. A data value represents one variable in an observation.

data value
in SAS, a unit of character or numeric information in a SAS data set. A data value represents one variable in an observation.

database
an organized collection of related data. A database usually contains named files, named objects, or other named entities such as tables, views, and indexes

database management system (DBMS)
an organized collection of related data. A database usually contains named files, named objects, or other named entities such as tables, views, and indexes

editing data
the process of viewing the contents of a file with the intent and the ability to change those contents. Depending on how the file is accessed, you can view the data either one observation at a time or in a tabular format.

engine
a component of SAS software that reads from or writes to a file. Each engine enables SAS to access files that are in a particular format. There are several types of engines.

file
a collection of related records that are treated as a unit. SAS files are processed and controlled by SAS and are stored in SAS data libraries.

format
a collection of related records that are treated as a unit. SAS files are processed and controlled by SAS and are stored in SAS data libraries. In SAS/ACCESS software, the default formats vary according to the interface product.

index
(1) in SAS software, a component of a SAS data set that enables SAS to access observations in the SAS data set quickly and efficiently. The purpose of SAS indexes is to optimize WHERE-clause processing and to facilitate BY-group processing. (2) in other software vendors' databases, a named object that directs the DBMS to the storage location of a particular data value for a particular column. Some DBMSs have additional specifications. These indexes are also used to optimize the processing of WHERE clauses and joins. Depending on the SAS interface to a database product and how selection criteria are specified, SAS may or may not be able to use the indexes of the DBMS to speed data retrieval.

Depending on how selection criteria are specified, SAS might use DBMS indices to speed data retrieval.

informat
a pattern or set of instructions that SAS uses to determine how data values in an input file should be interpreted. SAS provides a set of standard informats and also enables you to define your own informats.

interface view engine
a SAS engine that is used by SAS/ACCESS software to retrieve data from files that have been formatted by another vendor's software. Each SAS/ACCESS interface has its own interface view engine, which reads the interface product data and returns the data in a form that SAS can understand (that is, in a SAS data set). SAS automatically uses an interface view engine; the engine name is stored in SAS/ACCESS descriptor files so that you do not need to specify the engine name in a LIBNAME statement.

libref
a name that is temporarily associated with a SAS data library. The complete name of a SAS file consists of two words, separated by a period. The libref, which is the first word, indicates the library. The second word is the name of the specific SAS file. For example, in VLIB.NEWBDAY, the libref VLIB tells SAS which library contains the file NEWBDAY. You assign a libref with a LIBNAME statement or with an operating system command.

member
a SAS file in a SAS data library.

member name
a name that is given to a SAS file in a SAS data library.

member type
a SAS name that identifies the type of information that is stored in a SAS file. Member types include ACCESS, DATA, CATALOG, PROGRAM, and VIEW.

missing value
in SAS, a term that describes the contents of a variable that contains no data for a particular row or observation. By default, SAS prints or displays a missing numeric value as a single period, and it prints or displays a missing character value as a blank space.

observation
a row in a SAS data set. All of the data values in an observation are associated with a single entity such as a customer or a state. Each observation contains one data value for each variable. In a database product table, an observation is analogous to a row. Unlike rows in a database product table or file, observations in a SAS data file have an inherent order.

Pass-Through Facility
a group of SQL procedure statements that send and receive data directly between a relational database management system and SAS. The Pass-Through Facility includes the CONNECT, DISCONNECT, and EXECUTE statements, and the CONNECTION TO component. SAS/ACCESS software is required in order to use the Pass-Through Facility.

PROC SQL view
a SAS data set (of type VIEW) that is created by the SQL procedure. A PROC SQL view contains no data. Instead, it stores information that enables it to read data values from other files, which can include SAS data files, SAS/ACCESS views, DATA step views, or other PROC SQL views. A PROC SQL view's output can be either a subset or a superset of one or more files.

query
a set of instructions that requests particular information from one or more data sources.

referential integrity
a set of rules that a DBMS uses to ensure that whenever a data value in one table is changed, the appropriate change is also made to any related values in other tables or in the same table. Referential integrity is also used to ensure that related data is not deleted or changed accidentally.

relational database management system
a database management system that organizes and accesses data according to relationships between data items. Oracle and DB2 are examples of relational database management systems.

rollback
in most databases, the process that restores the database to its state when changes were last committed, voiding any recent changes. The SQL ROLLBACK statement initiates the rollback processes. See also commit.

row
in relational database management systems, the horizontal component of a table. A row is analogous to a SAS observation.

SAS data file
a type of SAS data set that contains data values as well as descriptor information that is associated with the data. The descriptor information includes information such as the data types and lengths of the variables, as well as the name of the engine that was used to create the data. A PROC SQL table is a SAS data file. SAS data files are of member type DATA.

SAS data library
a collection of one or more SAS files that are recognized by SAS and that are referenced and stored as a unit. Each file is a member of the library.

SAS data set
a file whose contents are in one of the native SAS file formats. There are two types of SAS data sets: SAS data files and SAS data views. SAS data files contain data values in addition to descriptor information that is associated with the data. SAS data views contain only the descriptor information plus other information that is required for retrieving data values from other SAS data sets or from files whose contents are in other software vendors' file formats.

SAS data view
a file whose contents are in one of the native SAS file formats. There are two types of SAS data sets: SAS data files and SAS data views. SAS data files contain data values in addition to descriptor information that is associated with the data. SAS data views contain only the descriptor information plus other information that is required for retrieving data values from other SAS data sets or from files whose contents are in other software vendors' file formats.

SAS/ACCESS views
See view descriptor and SAS data view.

server
in a network, a computer that is reserved for servicing other computers in the network. Servers can provide several different types of services, such as file services and communication services. Servers can also enable users to access shared resources such as disks, data, and modems.

Structured Query Language (SQL)
the standardized, high-level query language that is used in relational database management systems to create and manipulate database management system objects. SAS implements SQL through the SQL procedure.

table
a two-dimensional representation of data, in which the data values are arranged in rows and columns.

trigger
a type of user-defined stored procedure that is executed whenever a user issues a data-modification command such as INSERT, DELETE, or UPDATE for a specified table or column. Triggers can be used to implement referential integrity or to maintain business constraints.

variable
a column in a SAS data set. A variable is a set of data values that describe a given characteristic across all observations.

view
a definition of a virtual data set. The definition is named and stored for later use. A view contains no data; it merely describes or defines data that is stored elsewhere. SAS data views can be created by the ACCESS and SQL procedures.

view descriptor
a file created by SAS/ACCESS software that defines part or all of the database management system (DBMS) data or PC file data that is described by an access descriptor. The access descriptor describes the data in a single DBMS table, DBMS view, or PC file.

wildcard
a file created by SAS/ACCESS software that defines part or all of the database management system (DBMS) data or PC file data that is described by an access descriptor. The access descriptor describes the data in a single DBMS table, DBMS view, or PC file.

Index

S

T

U

V

Your Turn

We want your feedback.

- If you have comments about this book, please send them to **yourturn@sas.com**. Include the full title and page numbers (if applicable).
- If you have comments about the software, please send them to **suggest@sas.com**.