SAS® 9.1.3 Metadata LIBNAME Engine

User's Guide

Second Edition

The Power to Know.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2006. *SAS® 9.1.3 Metadata LIBNAME Engine: User's Guide, Second Edition*. Cary, NC: SAS Institute Inc.

SAS® 9.1.3 Metadata LIBNAME Engine: User's Guide, Second Edition

Copyright © 2006, SAS Institute Inc., Cary, NC, USA

ISBN-13: 978–1–59047–822–6
ISBN-10: 1–59047–822–3

All rights reserved. Produced in the United States of America.

For a hard-copy book: No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

For a Web download or e-book: Your use of this publication shall be governed by the terms established by the vendor at the time you acquire this publication.

U.S. Government Restricted Rights Notice. Use, duplication, or disclosure of this software and related documentation by the U.S. government is subject to the Agreement with SAS Institute and the restrictions set forth in FAR 52.227-19 Commercial Computer Software-Restricted Rights (June 1987).

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

1st printing, March 2006

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.

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

Contents

What's New

Overview

The metadata engine provides secure access to SAS data. By incorporating metadata, this engine augments and controls access to the data.

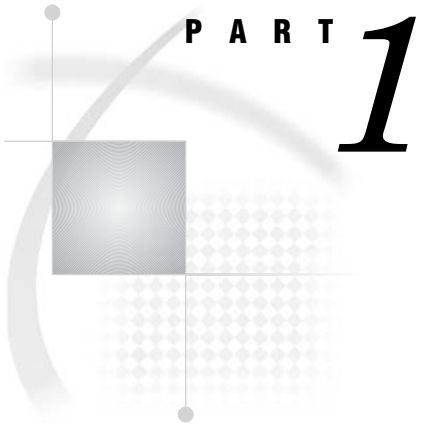
Note: This section describes the features of the SAS metadata engine that are new or enhanced since SAS 9.0. △

Details

- The metadata engine, in conjunction with the SAS Open Metadata Architecture Authorization Facility, enables an administrator to specify user privileges in order to control which data the users are allowed to access. See Chapter 7, "Metadata Requirements for Using the SAS Open Metadata Architecture Authorization Facility to Control Data Access," on page 41.
- The new LIBURI= and LIBRARY= options in the LIBNAME statement provide additional methods for referencing the SASLibrary metadata object. See the LIBURI= and LIBRARY= options in "LIBNAME Statement Syntax" on page 53.
- The new METAOUT= option specifies access to tables in the data source.
  - The METAOUT= option in the LIBNAME statement enables you to specify access to a library. See the METAOUT= option in "LIBNAME Statement Syntax" on page 53.
  - The METAOUT= data set option enables you to specify access to a specific table. See "METAOUT= Data Set Option" on page 59.
  - Beginning with SAS 9.1.3 Service Pack 4, with the default behavior of the engine (METAOUT=ALL), you can no longer create or delete tables or their associated metadata. You can insert, update, and delete observations in a table. Metadata must be present for you to access the table.
  - Beginning with SAS 9.1.3 Service Pack 4, when METAOUT=DATA is specified for a table, the metadata engine behaves more like the underlying engine that is defined in the metadata; for example, data set options can be passed directly to the underlying engine.

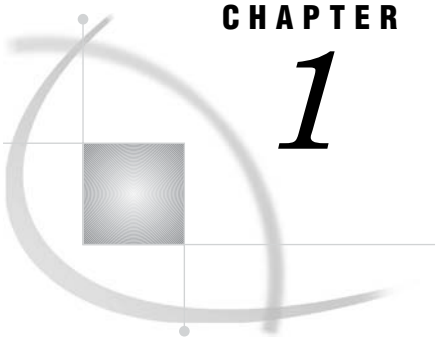
- Beginning with SAS 9.1.3 Service Pack 4, you are advised to use the METALIB procedure instead of METAOUT=META to create, update, or delete metadata. The METAOUT=META value is not expected to be supported in the next release of the software. (For information about PROC METALIB, see the chapter about SAS language metadata interfaces in *SAS Open Metadata Interface: Reference*).

- Beginning with SAS 9.1.3 Service Pack 4, SQL implicit pass-through is supported.



PART 1

Usage



CHAPTER

1

Using the Metadata LIBNAME Engine

What Does the Metadata Engine Do?

The metadata engine enables you to use metadata in order to process and augment data that is identified by the metadata. The metadata engine retrieves information about a target SAS data library from metadata objects in a specified SAS Metadata Repository.

The *metadata* is information about the structure and content of data and the applications that process and manipulate that data. The metadata contains details such as the location of the data and the SAS engine that is used to process the data.

The metadata engine provides a consistent method for accessing many data sources. For example, because SAS provides different engines, with different options, behavior, and tuning requirements, it can be difficult to keep track of how each SAS engine works. By taking advantage of metadata, the necessary information required to access data can be created in one central location so that applications can simply use the metadata engine in order to access different sources of data, without having to understand the differences and details of each SAS engine.

Understanding How the Metadata Engine Works

The metadata engine works much like other SAS engines. That is, you execute a LIBNAME statement in order to assign a libref and to specify an engine. You then use that libref throughout the SAS session where a libref is valid.

However, instead of the libref being associated with the physical location of a SAS data library, the metadata libref is associated with a set of metadata objects. The metadata objects identify the SAS engine that provides access to the data and options that are necessary to process the SAS data library and its members.

Here is an example of a LIBNAME statement for the metadata engine and a description of what happens when you execute the statement:

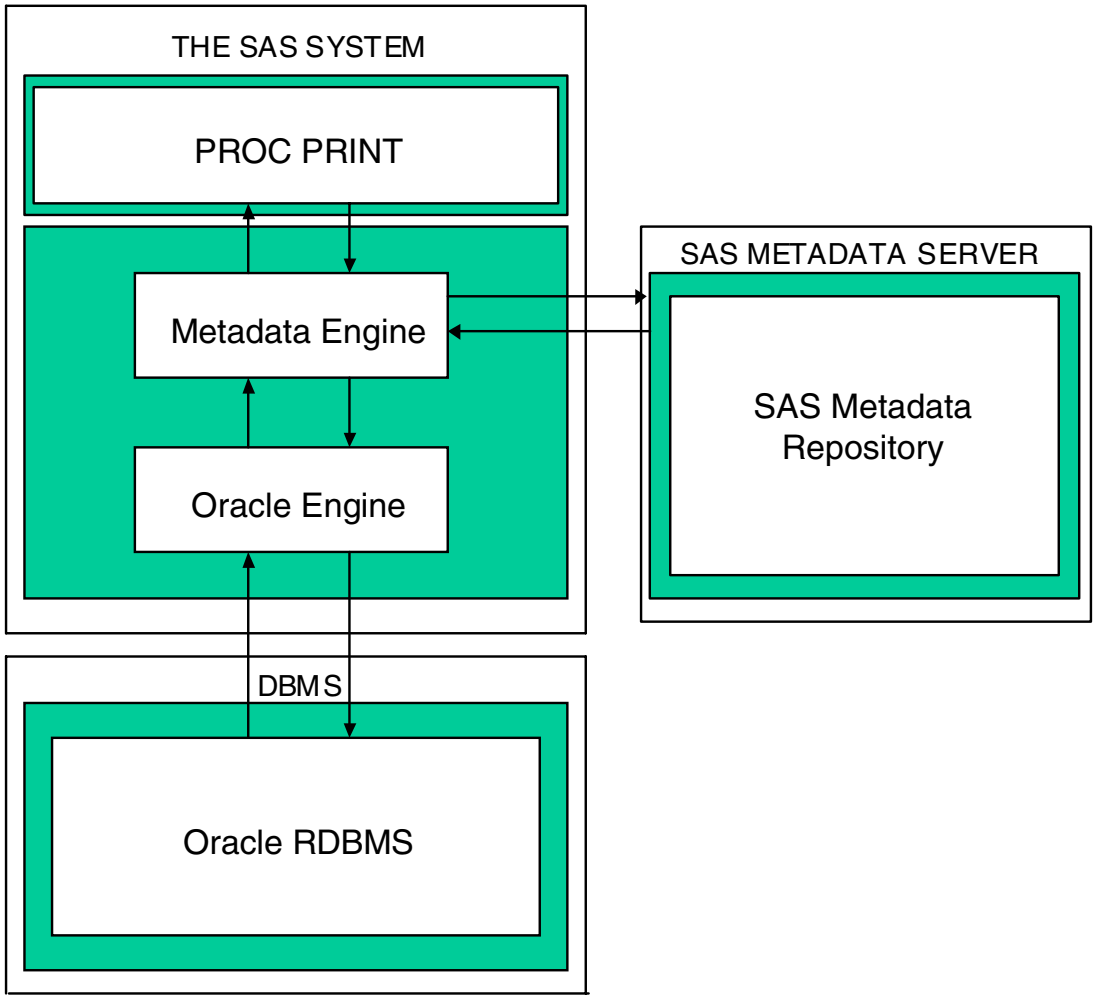
```
libname oralib meta libid=A8000001 repid=AWPKT800
   userid=metaid pw=metapw
   ipaddr=myip.us.org.com port=6401
   protocol=bridge liboptset=myopts;
```

1. The metadata engine retrieves information about the target SAS data library from the metadata.
2. The metadata engine uses the retrieved information to construct a LIBNAME statement for the engine that is specified in the metadata (referred to as the underlying engine) and assigns it the appropriate options.
3. Then, when the metadata engine needs to access data, the metadata engine uses the underlying engine to process the data.

For example, if you have an Oracle data library defined in metadata and you reference the library with the LIBNAME statement for the metadata engine, the metadata engine constructs a LIBNAME statement that assigns the Oracle SAS/ACCESS engine. When the metadata engine needs to process a member of the library, such as with the PRINT procedure, the metadata engine issues a request to the metadata repository for the metadata that is associated with that member and uses the Oracle engine. The metadata includes PhysicalTable, Column, and Property objects.

The following diagram illustrates the metadata engine process.

Figure 1.1 Metadata Engine Process



Advantages of Using the Metadata Engine

Using the metadata engine provides the following advantages:

- The metadata engine provides a single point of access to many heterogeneous data sources. You do not have to be aware of any of the engine-specific details.
- The metadata engine, in conjunction with the SAS Open Metadata Architecture Authorization facility, enables an administrator to control what data users are allowed to access in three ways:
  - The administrator can associate authorization metadata to any metadata resource in a repository using the SAS Open Metadata Architecture Authorization Facility. The metadata engine enforces the decision returned by the authorization facility for libraries and tables.
  - The administrator can include only those libraries, tables, and columns that users are allowed to access in the metadata. In this way, only those libraries, tables, and columns can be accessed by the metadata engine. For example, if a data source has five tables but only two are defined in the repository, the metadata engine provides access to only those two tables. In another situation, if a table has 20 columns but only five columns are defined in the

repository, the metadata engine provides access to only those five columns. In this context, the metadata engine acts as a filter, providing member- and column-level security.

- An administrator can associate authorization metadata to a metadata resource that prevents the user from accessing the resource in the data source. For example, if a library has 20 tables and all 20 tables are defined in the repository, then the administrator can associate authorization metadata with five tables, which prevents the user from accessing those tables. Therefore, the user will have access to only 15 tables. Again, the metadata engine acts as a filter, providing member-level security.

An advantage of this behavior is that an administrator can use the metadata engine as a means to provide library and table security. See Chapter 7, "Metadata Requirements for Using the SAS Open Metadata Architecture Authorization Facility to Control Data Access," on page 41.

- The SAS Open Metadata Architecture, which is a general-purpose metadata management facility that provides common metadata services to SAS applications, provides a permanent storage location to maintain column formats, informats, labels, and other information that cannot currently be saved with some data sources.
- Changing the type of data source you use does not require changing SAS code or other applications. SAS Open Metadata Architecture provides a centralized location for storing information about your data environment. Because the metadata engine uses this environment to access your data, you are insulated from these changes. It does not matter if your data changes from Oracle to DB2 to SAS; the SAS code remains the same.

What Is Supported?

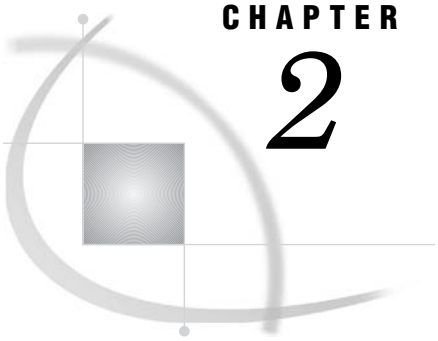
The metadata engine supports the following:

- The metadata engine accesses metadata in a SAS Metadata Repository in order to process data that the metadata identifies. Using the metadata engine and SAS code, you can have full read and write access (input, update, output) to tables in the data source.

  Note: The metadata engine supports only tables (SAS data sets and DBMS tables). The metadata engine does not support other SAS files such as catalogs or views. △

  See Chapter 3, "Examples of Using the Metadata Engine," on page 11.

- The metadata engine in conjunction with the SAS Open Metadata Architecture Authorization facility enables an administrator to control what data users are allowed to access.
- The metadata engine can augment your data. For example, if you execute the DATASETS procedure in order to get a listing of all the members in a library, the metadata engine sends a request to the metadata repository for this information. The metadata engine does not use the underlying engine. You get a listing of only those members that have been populated in the repository. When you execute the CONTENTS procedure, the table and column attributes that are returned are from the repository, not a result of the underlying engine processing the data. Any formats, informats, or labels that are stored in the metadata are applied to the underlying data. Any columns that are not populated in metadata are not available to the user.
- The metadata engine supports passing data set options directly to the underlying engine. LIBNAME statement options and data set options can also be placed in the SAS Metadata Repository to be used by the engine.
- The metadata engine supports SQL implicit pass-through.
- The metadata must be consistent with the identified data source. For example, it is invalid to have metadata for a SAS data set where a variable is defined as numeric data, but the actual data source defines it as character data.



CHAPTER

2

Understanding the Metadata

What Metadata Is Used by the Metadata Engine?

The metadata engine uses metadata that is stored in a specific SAS Metadata Repository on the SAS Metadata Server. The SAS Metadata Server provides metadata management services to one or more client applications. A SAS Metadata Repository is a collection of related metadata objects, such as the metadata for a set of tables and columns that are maintained by an application.

There are several methods to create metadata in a SAS Metadata Repository. For example, from the SAS Management Console, you can use a Library wizard in order to create the metadata objects in a repository that are necessary for the metadata engine to construct a LIBNAME statement. SAS Data Integration Studio enables you to define the table that you want to be a member in a library and any options that you want associated with that table. You provide information to SAS Data Integration Studio, and SAS Data Integration Studio generates corresponding metadata.

What Is Required to Use the Metadata Engine?

To use the metadata engine, the following is required:

- For the metadata engine to process data, the metadata must be available from an existing SASLibrary metadata object in a SAS Metadata Repository and must conform to specific metadata engine model requirements. See Chapter 4, “SAS Metadata Model Requirements for the Metadata Engine,” on page 17.
- For the metadata engine to access members in a SAS data library, the SASLibrary object must have an associated DatabaseSchema object for a DBMS SAS/ACCESS engine or a Directory object for a Base SAS data library.
- For the metadata engine to construct a LIBNAME statement for the underlying engine, appropriate metadata must be available. See Chapter 5, “Metadata Requirements to Construct the LIBNAME Statement for the Underlying Engine,” on page 23.
- To use the SAS Open Metadata Architecture Authorization Facility, authorization metadata can be defined that controls both the availability of specific metadata (ReadMetadata permission) as well as the actions that can be taken on the resource that a metadata object describes (Read, Write, Create, and Delete

permissions). See Chapter 7, “Metadata Requirements for Using the SAS Open Metadata Architecture Authorization Facility to Control Data Access,” on page 41.

- Metadata does not support some character data such as the ampersand (&) character. If the data that you want to add to a repository includes special characters like & (for example, in a column name or a label), you must represent the character as **`&`**. For example, if a table named Customer has the label **`My Account Names & Addresses`**, you must change it to **`My Account Names &amp; Addresses`**.

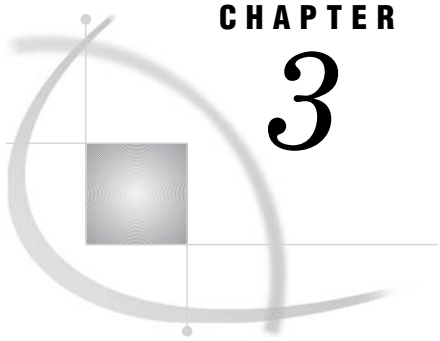
For information on the SAS Open Metadata Architecture and a SAS Metadata Repository, see

- *Getting Started with the SAS Open Metadata Interface*
- *SAS Open Metadata Interface: User's Guide*
- *SAS Open Metadata Interface: Reference*.

For information on the SAS Metadata Server, see

- *SAS Metadata Server: Setup and Administration Guide*

which is available from the SAS Community **`support.sas.com/rnd/eai/openmeta`**.



CHAPTER

3

Examples of Using the Metadata Engine

Submitting the LIBNAME Statement for the Metadata Engine

This example shows two LIBNAME statements—one statement that uses defaults and one statement that specifies all of the LIBNAME statement options.

- The following LIBNAME statement uses all defaults. The metadata server connection information is obtained from the metadata server system options, and the default set of options are used:

```
libname metaeng meta libid=AC000001;
```

- This example specifies all of the LIBNAME statement options for the metadata engine in order to connect to the metadata server. It also specifies particular PropertySet objects:

```
libname metaeng meta libid=AC000001
   repid=AA32V87R9 ipaddr='d5112.us.sas.com' port=6789 protocol=bridge
   userid=sasxyz pw=abcdefg
   liboptset='libset2' conoptset='conset2';
```

Accessing Data Using an Engine Directly Compared to Using the Metadata Engine

Overview

This example compares the process of accessing data using the Oracle SAS/ACCESS engine versus accessing the same data using the metadata engine. The goal is to access data that resides in an Oracle database, determine which tables exist in the data source, and print the contents of one of the tables.

Using the Oracle SAS/ACCESS Engine Directly

First, using the Oracle SAS/ACCESS engine directly to access the data, you would submit the following statements, which require that you know how to use the Oracle engine and that you know the options that are needed to access the data:

```
libname oralib oracle user=myuser pw=mypw
   path=ora_dbms preserve_tab_names=yes
   connection=sharedread schema=myschema; ❶

proc datasets library=oralib; ❷
quit;

proc print data=oralib.Sales (readbuff=1000); ❸
run;

data work.temp;
   set oralib.Sales (dbindex=myindex); ❹
run;
```

1 Identifies an Oracle library that contains the Oracle tables that you want to process.

2 Lists all of the Oracle tables that are available.

3 Displays the Oracle Sales table.

4 Attempts to use the specified index to improve performance.

Using the Metadata Engine

You can access the same data using the metadata engine. However, using the metadata engine, you do not have to know that you are using an Oracle database, and you do not have to know any of the Oracle SAS/ACCESS engine details.

Using the SAS Management Console and SAS Data Integration Studio, you create metadata in a repository for your Oracle environment. The metadata engine will interpret this metadata and locate your data. In addition, you do not have to know how to connect to the repository or know the values for the connection options. (These details can be provided using the metadata system options.)

Here is what happens when you use the metadata engine in order to access the Oracle data:

1 You submit the following LIBNAME statement for the metadata engine. LIBID= is the unique identifier of the SASLibrary object that defines information about the Oracle library and serves as an anchor point for obtaining other metadata. LIBID= is a required argument.

```
libname metaeng meta libid=A8000001 repid=AWPKT800
   userid=metaid pw=metapw
   ipaddr=myip.us.org.com port=6401
   protocol=bridge liboptset=myopts;
```

2 Using the information acquired from the LIBNAME statement, the metadata engine queries the repository that is specified by the REPID= value, on the server that is specified by the IPADDR= and PORT= values. The query results in information about the SASLibrary object that is specified by the LIBID= and other objects that are associated with the SASLibrary metadata object such as DatabaseSchema, SASClientConnection, Login, or Property objects.

Below are the objects that are returned from the query. The objects that appear display only those attributes that are used by the metadata engine and do not include the SASClientConnection objects and PropertySet objects that are included in the model.

Table 3.1 SASLibrary Object Query Results

SASLibrary	DatabaseSchema	Login	Property	Property	Property
Libref: oralib	*SchemaName:* myschema	*Userid:* myuser	*PropertyName:* preserve_tab_ names	*PropertyName:* path	*PropertyName:* Connection
Engine: oracle		*Password:* mypw	*DefaultValue:* yes	*DefaultValue:* ora_dbms	*DefaultValue:* sharedread
isDBMSLibname: TRUE			*Delimiter:* =	*Delimiter:* =	*Delimiter:* =
			UseValueOnly: False	*UseValueOnly:* False	UseValueOnly: False

Note: See the relational DBMS model in Chapter 4, "SAS Metadata Model Requirements for the Metadata Engine," on page 17 in order to determine how to associate these objects with one another. △

3 From the information embedded in the objects and from information that is implied in the relational DBMS model, the metadata engine is able to generate the following LIBNAME statement, which is the same LIBNAME statement that is shown at the beginning of this example:

```
libname oralib oracle user=myuser pw=mypw
   path=ora_dbms preserve_tab_names=yes
   connection=sharedread schema=myschema;
```

4 With the generated LIBNAME statement, the metadata engine uses the Oracle engine anytime it needs to access the Oracle data. For example, to view the tables that exist, you would submit

```
proc datasets library=metaeng;
quit;
```

The metadata engine sends a query to the repository that requests all members of the SASLibrary that were specified by LIBID=. The metadata engine returns only those members that are defined in the repository. Any Oracle table that does not exist in the metadata is not displayed. A benefit of this behavior is that administrators can control which data their users can access. Any attempt to use a member that does not have corresponding metadata in the repository will return an error.

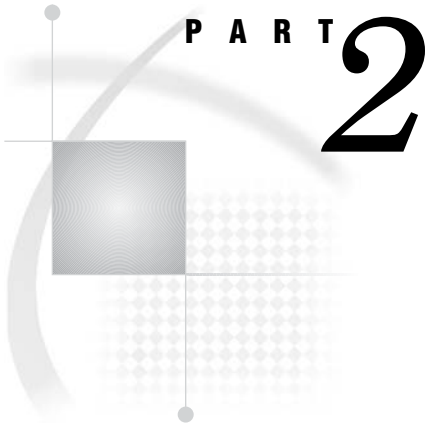
5 For the following PRINT procedure, the metadata engine sends a request to the repository for the metadata that is associated with the particular table. The metadata includes PhysicalTable, Column, and Property object information. The OPTSET= data set option, a metadata engine option, tells the metadata engine to return those properties that are a part of the MYOPTS property set. Within this property set are data set options that have been customized for this table. These data set options are used by the Oracle engine while processing the data:

```
proc print data=metaeng.Sales (optset=myopts);
run;
```

The metadata engine returns those columns that are defined in the metadata. Therefore, if the Oracle table Sales has 20 columns and only five columns are defined in the metadata, then you see only five columns. This is another way in which an administrator can use the metadata engine in order to control access to data.

6 The relational DBMS model allows you to store index information for given tables. Each Index object is associated with a table and with the columns that make up the index. Any use of the metadata engine that uses indexes results in a query to the repository that requests index information. The index metadata must match the physical index on the table. In the following statements, the OPTSET= data set option specifies to use indexes to process the table. The metadata engine uses the index information that is stored in the repository:

```
data work.temp;
   set metaeng.Sales (optset=index_opts);
run;
```



PART 2

Metadata Requirements

CHAPTER

4

SAS Metadata Model Requirements for the Metadata Engine

Understanding the Model Requirements

The SAS Metadata Model provides classes and objects that define different types of application metadata that is stored in a repository. The metadata engine uses the SAS Metadata Model as a framework and a common format for metadata modeling. In order for the metadata engine to access metadata objects that are stored in a metadata repository, the library metadata must be configured so that the metadata engine can process it.

The metadata engine supports the following models:

Relational DBMS Model	models a DBMS library, which uses a DBMS SAS/ACCESS engine.
SAS Data Set Model	models a Base SAS data library, which uses a Base SAS engine.
Remote Relational DBMS Model	models a remote DBMS library, which uses a SAS/ACCESS engine and the remote engine.
Remote SAS Data Set Model	models a remote Base SAS data library, which uses a Base SAS engine and the remote engine.

The following diagrams illustrate the associations between related metadata types in each supported model. The purpose of the diagrams is to help you understand the relationships among the metadata types in the SAS namespace.

Figure 4.1 Relational DBMS Model

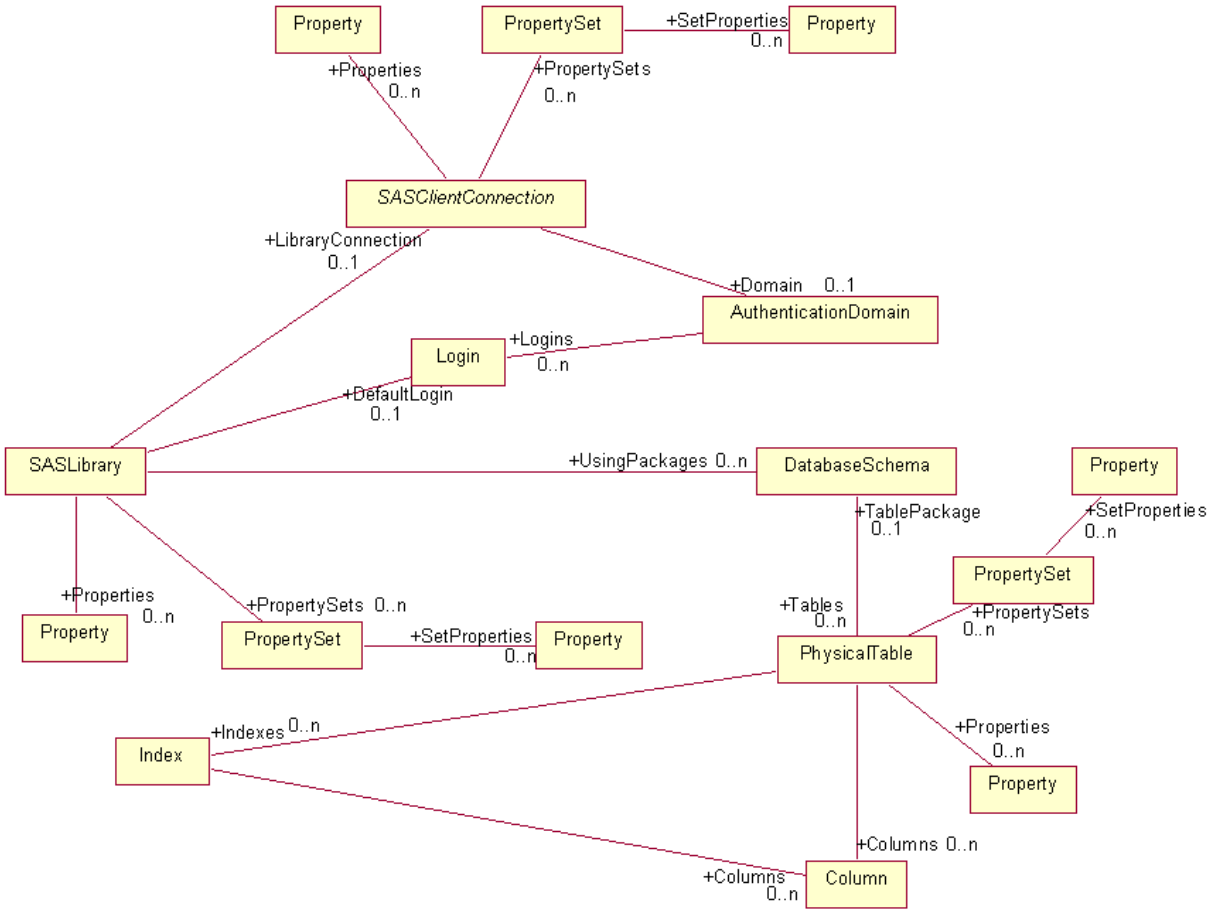


Figure 4.2 SAS Data Set Model

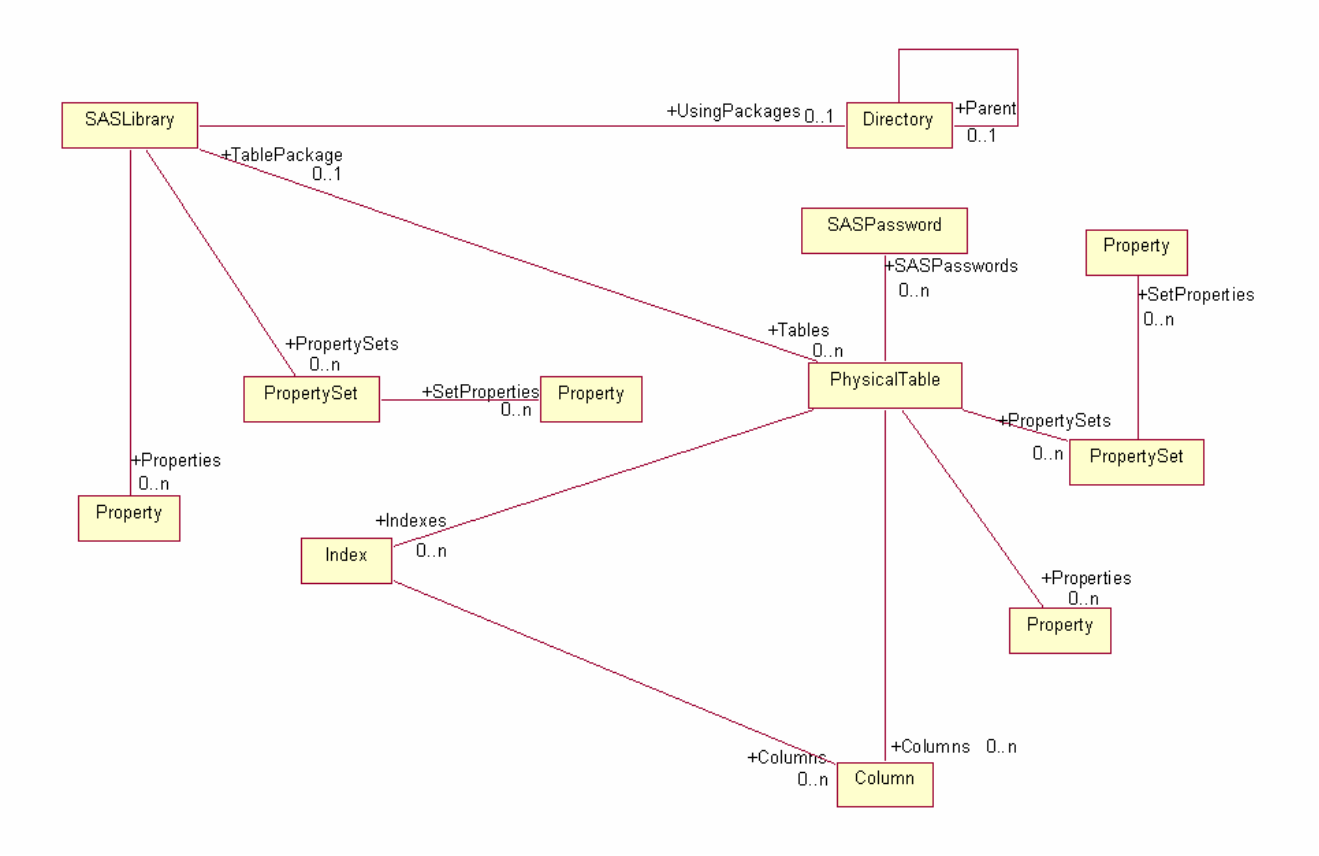


Figure 4.3 Remote Relational DBMS Model

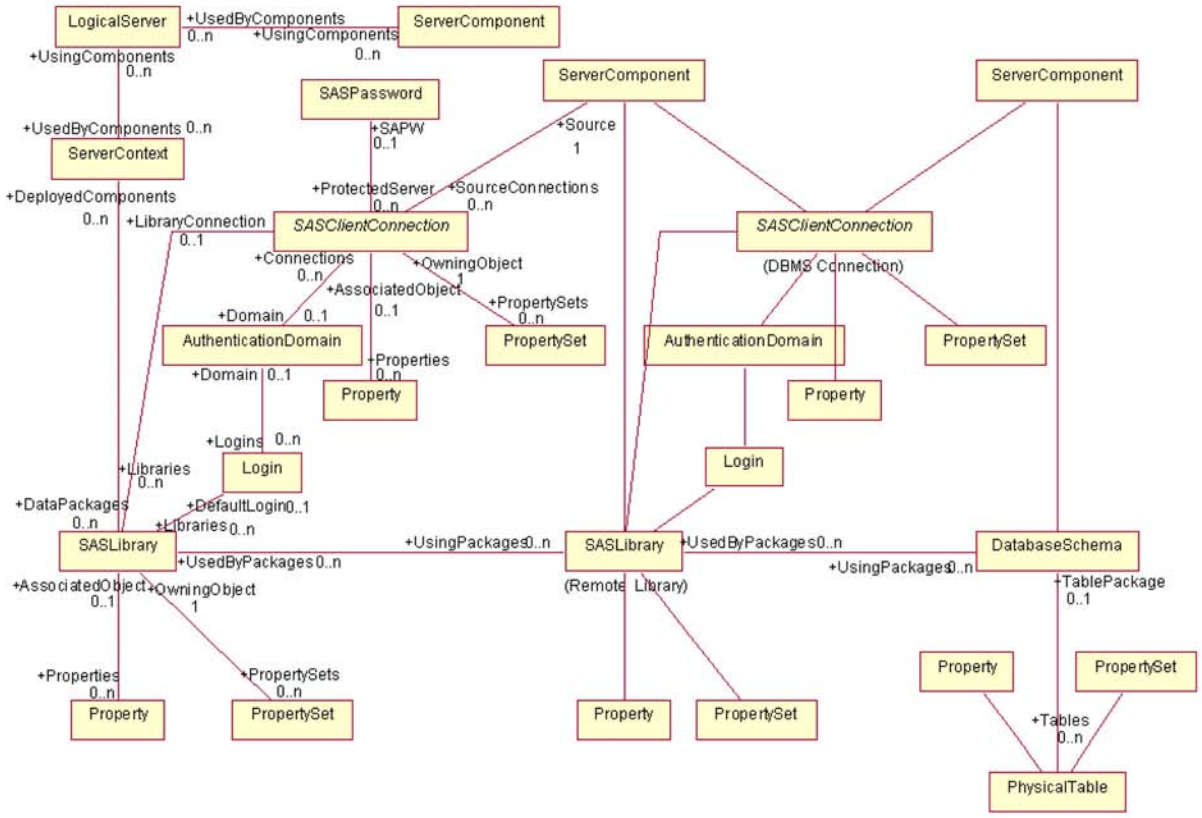
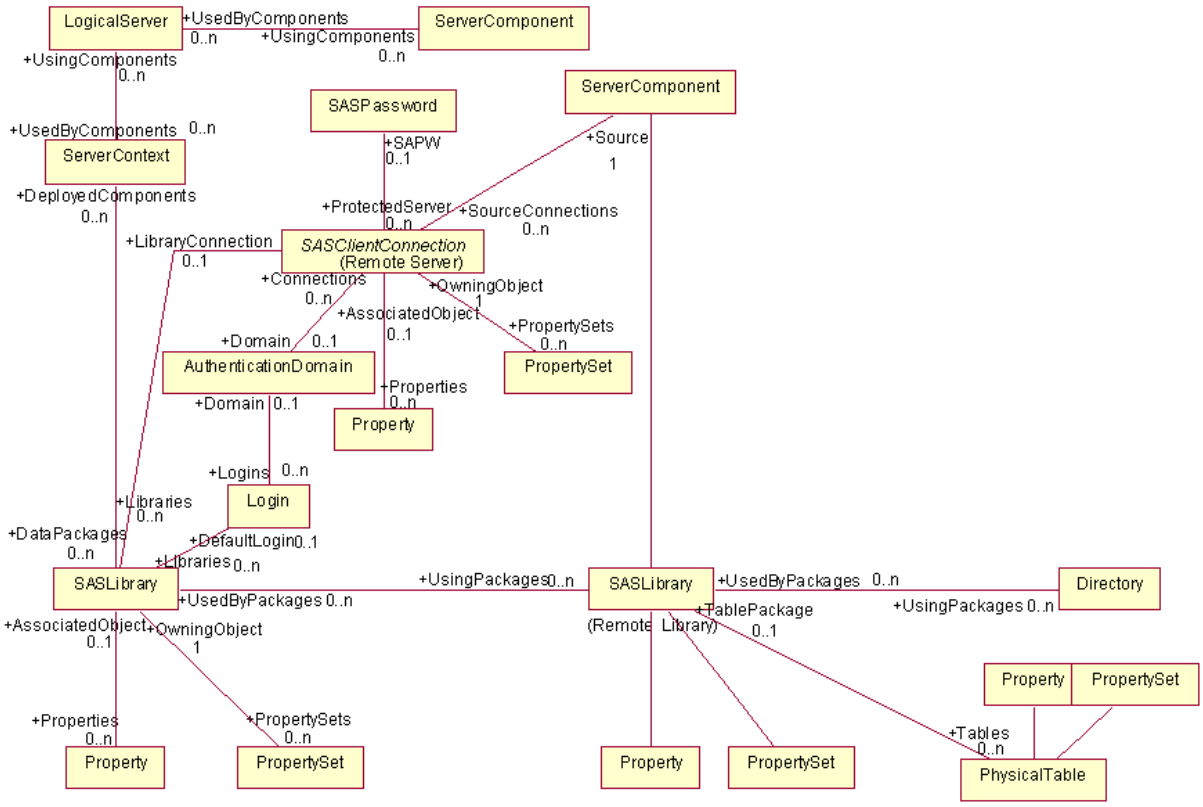


Figure 4.4 Remote SAS Data Set Model



CHAPTER 5

Metadata Requirements to Construct the LIBNAME Statement for the Underlying Engine

Understanding How the Metadata Engine Constructs a LIBNAME Statement

When you submit a LIBNAME statement for the metadata engine, the engine constructs a LIBNAME statement for a SAS data library. The primary object is the SASLibrary object specified by the LIBID=, LIBURI=, or LIBRARY= argument. Any other object that the metadata engine needs in order to construct the LIBNAME statement is associated directly or indirectly with the SASLibrary object.

The SASLibrary object represents a SAS data library and provides the metadata engine with information about which engine processes the library and how it should be assigned. The SASLibrary object contains the following attributes that are relevant to the metadata engine:

Libref	is the libref to be used when assigning the SAS data library.
Engine	is the engine name to be used when assigning the SAS data library.
IsDBMSLibname	indicates whether the SAS data library will be used with a DBMS engine.

Options, in SAS Open Metadata Architecture, are represented with Property objects. One or more Property objects can be associated directly with an owning object (a SASLibrary, SASClientConnection, or PhysicalTable object) as the default set of options. One or more Property objects can be grouped using a PropertySet object. One or more PropertySet objects can be associated with an owning object. Either the default set of options or one PropertySet object can be used with an owning object at a particular time. The LIBOPTSET=, CONOPTSET=, and OPTSET= metadata engine options are used to specify a PropertySet object. No metadata engine option is needed to use the default set of options.

The Property object contains the following attributes that are relevant to the metadata engine:

PropertyName	is the name of the option.
Delimiter	is the delimiter to be used between the option name and value, that is, =.

DefaultValue is the value of the option.

UseValueOnly indicates whether to use the option value as the option (a boolean option) or to use a name-value pair.

Metadata Requirements to Construct a LIBNAME Statement for a Base SAS Engine

In order for the metadata engine to construct a LIBNAME statement for a Base SAS engine as shown in the following syntax, the metadata must be available.

Note: The metadata engine constructs a LIBNAME statement based on the stored metadata and uses this LIBNAME statement in order to access your data. This operation is transparent; however, it is important to understand the components of the LIBNAME statement so that you can configure the metadata appropriately using the SAS Management Console. △

LIBNAME *libref* <*engine*> '*SAS-data-library*' <*options*>;

libref
is any valid libref, as documented in *SAS Language Reference: Dictionary*. The libref references the SAS data library that the Base SAS engine will process. The value for the libref is obtained from the Libref attribute in the SASLibrary object.

engine
is the name of the Base SAS engine that will process the SAS data library. The engine name is obtained from the Engine attribute in the SASLibrary object.

'*SAS-data-library*'
is the physical name for the SAS data library. In SAS Open Metadata Architecture, physical names are represented with Directory objects. The value is obtained from the DirectoryName attribute in the Directory object that is associated with the SASLibrary object. The DirectoryName attribute should include any operating environment specific delimiters. The Directory object contains the following attributes that are relevant to the metadata engine:

DirectoryName is the physical name for the directory or SASlibrary object.

IsRelative indicates whether the value in the DirectoryName attribute is the complete physical name or is relative to a parent directory.

If the DirectoryName attribute does not contain the complete physical name and is relative to a parent directory, the metadata engine will retrieve the parent directory. Then it will append the value in the DirectoryName attribute for the parent directory to the beginning of the value in the DirectoryName attribute of the subdirectory (the physical name is complete when the IsRelative attribute is set to false). There might be several parent directories involved before the physical name is completely constructed. After the metadata engine has constructed a complete physical name, it is included as the SAS data library argument in the LIBNAME statement for the Base SAS engine.

options
are the LIBNAME statement options for the Base SAS engine. These options are represented by groups of Property objects that are associated directly with the SASLibrary object. PropertySet objects are used to associate different groups of Property objects with the SASLibrary object. Each option is generated using the PropertyName, Delimiter, and DefaultValue attributes in a Property object.

The following table shows how options are represented by individual Property objects associated with a property set. The BASELIB PropertySet object is associated with a SASLibrary object.

Table 5.1 BASELIB PropertySet Object

SAS Option	PropertyName Attribute	Delimiter Attribute	DefaultValue Attribute	UseValueOnly Attribute
REPEMTY=NO	REPEMPTY	=	NO	0
ACCESS=READONLY	ACCESS	=	READONLY	0
NODLTRUNCHK			NODLTRUNCHK	1
EXTEND			EXTEND	1

For example, the metadata engine would construct the following LIBNAME statement for the Base SAS engine using the SAS Open Metadata Architecture metadata that is shown in the subsequent table. (The NODLTRUNCHK and EXTEND z/OS options are included only to show how Property objects represent boolean options.)

```
libname sas9 v9 'C:\sales' repempty=no access=readonly nodltrunchk extend;
```

Table 5.2 Metadata for Base SAS Engine LIBNAME Statement

Parameter	Value	Metadata Object	Object Attribute
libref	sas9	SASLibrary (unique identifier = AD000001)	Libref='sas9'
engine	V9	SASLibrary (same object as above)	Engine='V9'
'SAS-data-library'	sales	Directory	DirectoryName='sales' IsRelative='1'
'SAS-data-library'	C:\	Directory	DirectoryName='C:\' IsRelative='0'
options	repempty	Property	PropertyName='repempty' Delimiter='=' DefaultValue='no' UseValueOnly='0'
options	access	Property	PropertyName='access' Delimiter='=' DefaultValue='readonly' UseValueOnly='0'
options	nodltrunchk	Property	DefaultValue='nodltrunchk' UseValueOnly='1'
options	extend	Property	DefaultValue='extend' UseValueOnly='1'

The previous Base SAS engine LIBNAME statement would be constructed when one of the following LIBNAME statements for the metadata engine is submitted:

- This LIBNAME statement would be submitted if the default set of Property objects representing the REPEMPTY, ACCESS, NODLTRUNCHK, and EXTEND LIBNAME options are associated with the SASLibrary object:

```
libname y meta libid=AD000001 repname=sasrepos
   ipaddr='D5678.us.sas.com' port=1234
   userid=sasabc pw=srvpw;
```

- This LIBNAME statement would be submitted if the Property objects representing the REPEMPTY, ACCESS, NODLTRUNCHK, and EXTEND options are associated with the BASELIB PropertySet object:

```
libname y meta libid=AD000001 repname=sasrepos
   ipaddr='D5678.us.sas.com' port=1234
   userid=sasabc pw=srvpw liboptset=baselib;
```

Metadata Requirements to Construct a LIBNAME Statement for a DBMS SAS/ACCESS Engine

In order for the metadata engine to construct a LIBNAME statement for a DBMS SAS/ACCESS engine as shown in the following syntax, the metadata must be available.

Note: The metadata engine constructs a LIBNAME statement based on the stored metadata and uses this LIBNAME statement in order to access your data. This operation is transparent; however, it is important to understand the components of the LIBNAME statement so that you can configure the metadata appropriately using the SAS Management Console. △

LIBNAME *libref SAS/ACCESS-engine-name*<*SAS/ACCESS-engine-connection-options*> <*SAS/ACCESS-LIBNAME-options*>;

libref
is any valid libref, as documented in *SAS Language Reference: Dictionary*. The libref references the SAS data library that the SAS/ACCESS engine will process. The value for the libref is obtained from the Libref attribute in the SASLibrary object.

SAS/ACCESS-engine-name
is the name of the underlying engine that will process the SAS data library. The engine name is obtained from the Engine attribute in the SASLibrary object.

SAS/ACCESS-engine-connection-options
are the engine-specific connection options for the LIBNAME statement for the SAS/ACCESS engine. These options are represented by groups of Property objects that are associated directly with the SASClientConnection object associated with the SASLibrary object. PropertySet objects are used to associate different groups of Property objects with the SASClientConnection object. Each connection option is generated using the PropertyName, Delimiter, and DefaultValue attributes in a Property object.

Exception: The values for USERID= and PASSWORD= connection options are obtained from a Login object. The Login object can be associated with the SASLibrary object. If no Login object is associated with the SASLibrary object, a Login object is retrieved from the AuthenticationDomain that is associated with the SASClientConnection object.

SAS/ACCESS-LIBNAME-options
are the LIBNAME statement options for the SAS/ACCESS engine. These options are represented by groups of Property objects that are associated directly with the SASLibrary object. PropertySet objects are used to associate different groups of Property objects with the SASLibrary object. Each LIBNAME option is generated using the PropertyName, Delimiter, and DefaultValue attributes in a Property object.

Exception: The value for the SCHEMA= option is obtained from a DatabaseSchema object. This DatabaseSchema object must be associated with the SASLibrary object if the SAS data library will be processed by a DBMS engine.

The following tables show how SAS/ACCESS options are represented in individual Property objects associated with a property set. The READCON PropertySet object would be associated with a SASClientConnection object, and the READLIB PropertySet object would be associated with a SASLibrary object.

Table 5.3 READCON Connection PropertySet Object

SAS/ACCESS Option	PropertyName Attribute	Delimiter Attribute	DefaultValue Attribute	UseValueOnly Attribute
CONNECTION=SHAREDREAD	CONNECTION	=	SHAREDREAD	0
PATH=ORACLEV8	PATH	=	ORACLEV8	0

Table 5.4 READLIB LIBNAME PropertySet Object

SAS/ACCESS Option	PropertyName Attribute	Delimiter Attribute	DefaultValue Attribute	UseValueOnly Attribute
ACCESS=READONLY	ACCESS	=	READONLY	0
DBINDEX=YES	DBINDEX	=	YES	0

For example, the metadata engine would construct the following LIBNAME statement for the Oracle SAS/ACCESS engine using the SAS Open Metadata Architecture metadata that is shown in the subsequent table:

```
libname oralib user=scott password=tiger path=oraclev8
   schema=sales dbindex=yes
   connection=sharedread access=readonly;
```

Table 5.5 Metadata for SAS/ACCESS Engine LIBNAME Statement

Parameter	Value	Metadata Object	Object Attribute
libref	oralib	SASLibrary (Unique identifier = AC000001)	Libref='oralib'
SAS/ ACCESS-engine-name	oracle	SASLibrary (same object as above)	Engine = 'oracle'
SAS/ACCESS-engine-connection-option	user	Login	Userid='scott'
SAS/ACCESS-engine-connection-option	password	Login (same object as above)	Password='tiger'

Parameter	Value	Metadata Object	Object Attribute
SAS/ACCESS-engine-connection-option	path	Property	PropertyName='path' Delimiter='=' DefaultValue='oraclev8' UseValueOnly='0'
SAS/ACCESS-LIBNAME-option	schema	DatabaseSchema	SchemaName='sales'
SAS/ACCESS-LIBNAME-option	dbindex	Property	PropertyName='dbindex' Delimiter='=' DefaultValue='yes' UseValueOnly='0'
SAS/ACCESS-engine-connection-option	connection	Property	PropertyName='connection' Delimiter='=' DefaultValue='sharedread' UseValueOnly='0'
SAS/ACCESS-LIBNAME-option	access	Property	PropertyName='access' Delimiter='=' DefaultValue='readonly' UseValueOnly='0'

The previous SAS/ACCESS LIBNAME statement would be constructed when one of the following LIBNAME statements for the metadata engine is submitted:

- This LIBNAME statement would be submitted if the default set of Property objects representing the PATH and CONNECTION options is associated with the SASClientConnection object, and the default set of Property objects representing the DBINDEX and ACCESS LIBNAME options is associated with the SASLibrary object:

```
libname x meta libid=AC000001 repname=orarepos
   ipaddr='D5678.us.sas.com' port=1234
   userid=sasabc pw=srvpw;
```

- This LIBNAME statement would be submitted if the Property objects representing the PATH and CONNECTION connection options are associated with the READCON PropertySet object (shown above) and the Property objects representing the DBINDEX and ACCESS LIBNAME options are associated with the READLIB PropertySet object:

```
libname x meta libid=AC000001 repname=orarepos
   ipaddr='D5678.us.sas.com' port=1234
   userid=sasabc pw=srvpw
   conoptset=readcon liboptset=readlib;
```

Metadata Requirements to Construct a LIBNAME Statement for the Remote Engine

In order for the metadata engine to construct a LIBNAME statement for the remote engine as shown in the following syntax, the metadata must be available.

Note: The metadata engine constructs a LIBNAME statement based on the stored metadata and uses this LIBNAME statement in order to access your data. This operation is transparent; however, it is important to understand the components of the LIBNAME statement so that you can configure the metadata appropriately. △

LIBNAME *libref* SERVER=*serverid* <ACCESS=READONLY> <OUTREP=*format*> <USER=*userid*> <PASSWORD=*password*> <SAPW=*server-access-password*> <SLIBREF=*server-libref*> ;

libref
is any valid libref, as documented in *SAS Language Reference: Dictionary*. The value for the libref is obtained from the Libref attribute in the SASLibrary object that represents the local SAS data library. This value can be the same or different from the libref included in the SASLibrary object that represents the remote SAS data library.

SERVER=*serverid*
specifies a name for the SAS server. The option is obtained from the PropertyName, Delimiter, and DefaultValue attributes in a Property object that is associated with the client connection for the SAS server.

ACCESS=READONLY
controls read access to a data library, in this case, through the SAS server. The option is obtained from the PropertyName, Delimiter, and DefaultValue attributes in a Property object that is associated with the local SAS data library.

OUTREP=*format*
creates new files in a foreign host format. The option is obtained from the PropertyName, Delimiter, and DefaultValue attributes in a Property object that is associated with the local SAS data library.

USER=*username*
specifies the user ID of the accessing client on the server. The value is obtained from the Userid attribute in a Login object.

PASSWORD=*server-access password*
specifies the password for the accessing client on the server. The value is obtained from the Password attribute in a Login object.

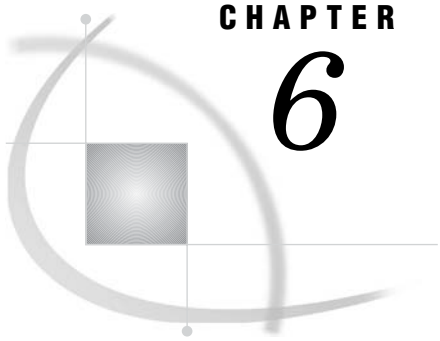
SAPW=*password*
specifies a server access password, which is used to gain access to the SAS server. The value is obtained from the Password attribute in a SASPassword object that is associated with the client connection for the SAS server.

SLIBREF=*server-libref*
specifies the libref that is used by the server to identify a SAS data library. The value is obtained from the Libref attribute in the SASLibrary object that represents the remote SAS data library.

For example, the following LIBNAME statements are for a remote library:

```
libname A server=BOB userid=TOM password=MLE sapw=alpha slibref=A;

libname B server=BOB userid=TOM password=MLE sapw=alpha slibref=A;
```



CHAPTER

6

Metadata Engine's Usage of the SAS Open Metadata Architecture SAS Namespace Types

What Is a SAS Namespace Type?

A SAS namespace defines metadata types for the most commonly used SAS applications. A namespace is a group of metadata types and their properties. Names are used to partition metadata into different contexts.

The SAS namespace contains metadata types for application elements such as SAS data sets and variables. A metadata type is a template that models the metadata for a particular object. For example, the metadata type Column models the metadata for a SAS data set variable (column), and the metadata type RepositoryBase models the metadata for a repository. Each metadata object is an instance of a metadata type, such as the metadata for a particular data store or the metadata for a particular repository.

How the Metadata Engine Uses SAS Namespace Types

The following describes how the metadata engine uses the SAS Open Metadata Architecture SAS namespace types that are configured in the models used by the metadata engine:

AuthenticationDomain
: represents the domain that controls user access to the server for the underlying engine. User login information will be retrieved from the authentication domain when no default login information has been provided with the SAS data library. This user login information will be included in the LIBNAME statement constructed for the underlying engine. An AuthenticationDomain object will not exist if user IDs and passwords are not used to connect to the server, or if a Base SAS engine is the underlying engine.

  Attributes: N/A

  Associations: *Logins* (0..n)

  This association from the AuthenticationDomain object to Login object(s) is used to retrieve only the user logins (Login objects) the user is authorized (via SAS Open Metadata Architecture security) to use. The logic for retrieving user login information is located under the description for the Login

object. The cardinality on this association enables an authentication domain to control several user logins.

Column
represents a column on a library member. Column metadata is retrieved when opening a library member and when retrieving indexes on an opened library member.

Attributes: The metadata engine uses the following attributes from the Column object:

- *Desc*—The SAS column label field.
- *SASColumnName*—The column name that will be used by SAS. This attribute must be populated.
- *SASColumnType*—The column type that will be used by SAS. This attribute must be populated.
- *SASColumnLength*—The column length that will be used by SAS. This attribute must be populated.
- *SASFormat*—The SAS format that will be applied to the column.
- *SASInformat*—The SAS informat that will be applied to the column.

Associations: N/A

DatabaseSchema
represents a DBMS schema. The SCHEMA= option cannot be represented with a Property object. The DBMS schema must be represented with a DatabaseSchema object. The DatabaseSchema object, if available, must be associated with a SASLibrary object. Only one schema can be associated with the SAS data library. If more than one schema is retrieved, the first one is used.

DBMS schema metadata is retrieved to obtain the name of the DBMS schema that will be included in the LIBNAME statement for the underlying engine. The DBMS schema metadata is retrieved after the LIBNAME statement for the metadata engine is executed. The DBMS schema is used as a point of reference when retrieving library member information and opening a library member.

Attributes: *SchemaName*—The name of the DBMS schema. The metadata engine will include this schema name as the value for the SCHEMA= option in the LIBNAME statement for the underlying engine. This attribute must be populated.

Exception: The SAS/ACCESS engines for ODBC, OLE DB, ACCESS, and EXCEL will not use a schema when their underlying data source does not support schemas. In these cases, this SchemaName attribute will be blank.

Associations: *Tables*(0..n)

This association from the DatabaseSchema object to PhysicalTable object(s) will be used to obtain all DBMS tables that are associated with the DBMS schema in a SAS Metadata Repository. The set of tables that is associated with the schema might differ from the tables that are under the schema in the DBMS. The cardinality on this association enables the DBMS schema to have many DBMS tables.

Directory
represents a file system directory. Directory metadata is retrieved after the LIBNAME statement for the metadata engine is executed. Directory metadata is

retrieved to obtain the physical name to be included as library specifications in the LIBNAME statement for the underlying engine. No library members are associated with a Directory object. The library members are associated with the SASLibrary object that is using this Directory object.

Attributes:

- *DirectoryName*—Directory name including any file system specific delimiters. The metadata engine treats the value of this attribute as a string, including the total string in the LIBNAME statement for the underlying engine.

  Note: For the metadata engine, do not use quotation marks in the directory path. △
- *IsRelative*—Indicates this directory is a subdirectory and has a parent directory. When this attribute is true, the DirectoryName attribute does not contain the complete name. The parent directory must be retrieved to complete the name. There could be several subdirectories involved in constructing a complete directory name.

Associations: *Parent*(0..1)

This association from the Directory object to another Directory object is used to obtain the remaining name when a directory is relative to another directory. In order to construct a complete directory name, the value of the DirectoryName attribute in the parent directory is appended to the beginning of the value of the DirectoryName attribute of the subdirectory. The cardinality on this association enables a subdirectory to have (at most) one parent directory. However, a chain of many subdirectories and parents might have to be traversed before the directory name is complete.

Index

represents an index on a library member. When index information is requested for an opened library member, index metadata is retrieved to obtain the name of the index and column(s) that comprise the index.

Attributes:

- *IndexName*—The name of the index on the library member. This attribute must be populated.

Associations:

- *Columns*(0..n)

  This association from the Index object to column object(s) will be used to obtain the columns that make up the index. The cardinality on this association enables the index to be a simple index (of one column) or a composite index (made up of more than one column).

Login

represents a user's login information (user ID/password) that will be used to log on to a server. This user login information is retrieved after the LIBNAME statement for the metadata engine is executed and will be included in the LIBNAME statement constructed for the underlying engine. When possible, a Login object should be associated with the SASLibrary object as the default login to use to connect to the server. If no Login object is associated with the SASLibrary object, the login information is retrieved via the AuthenticationDomain object.

The logic for retrieving user login information via the AuthenticationDomain object is as follows:

- If no Login object is retrieved for the user, no user login information will be included in the LIBNAME statement for the underlying engine. It will be

assumed that either no user login information is required to connect to the server for the underlying engine or the user login information has been stored in a location according to the setup for the underlying engine. A connection will be attempted without user login information and the user will be notified of any connection failures.

- If there is only one Login object retrieved, this user ID/password will be included in the LIBNAME statement for the underlying engine as values for the USERID= and PASSWORD= connection options.
- If there is more than one Login object retrieved, there is nothing to indicate which Login object to use. Therefore, no user login information will be included in the LIBNAME statement for the underlying engine.

Attributes: The following attributes must be populated in the Login object.

- *Userid*—A user's ID that will be used to log on to a server for the underlying engine.
- *Password*—The password that will enable a user to access a server for the underlying engine.

Associations: N/A

PhysicalTable

represents a member in a SAS data library. Metadata for a library member is retrieved when a member listing is requested, a library member is being opened for use, or when the use of an index has been requested. Metadata for a library member is retrieved in order to obtain the name and type of the member and, in some cases, the column on the library member.

Attributes:

- *Desc*—The SAS label field.
- *SASTableName*—The SAS name for a library member (that is, DBMS table, SAS data set, and so on). This attribute must be populated.
- *MemberType*—The SAS type for a library member (for example, DATA or VIEW). This attribute must be populated.

Associations:

- *Columns*(0..n)

  This association from the PhysicalTable object to Column object(s) will be used to obtain the columns on an opened library member. The cardinality on this association enables a library member to have many columns.

- *Indexes*(0..n)

  This association from the PhysicalTable object to Index object(s) will be used to identify the indexes on an opened library member. The cardinality on this association enables a library member to have several indexes.

- *Properties*(0..n)

  This association from the PhysicalTable object to Property object(s) will be used to obtain the data set options to be used when accessing a library member. These options will be considered the default set of data set options and will be applied each time the library member is opened unless a property set is specified in the metadata engine data set option OPTSET=. The

cardinality on this association enables a library member to have an unlimited number of data set options.

- *PropertySets*(0..n)

  This association from the PhysicalTable object to PropertySet object(s) is used to obtain a set of data set options to be used when opening a library member. This association is used when a property set name is specified in the metadata engine data set option OPTSET=. If OPTSET= is not specified, any default data set options from the Properties association is applied. The cardinality on this association enables a library member to have many property sets.

- *SASPasswords*(0..n)

  This association from the PhysicalTable object to SASPassword object(s) is used to obtain the READ=, WRITE=, ALTER=, and PW= passwords for a SAS data set. This association is valid only when a Base SAS engine is assigned as the underlying engine for the metadata engine. This association is used each time the SAS data set is opened for use. The cardinality on this association enables a SAS data set to have more than one SAS password.

- *TablePackage*(1)

  This association from the PhysicalTable object to a DatabaseSchema or SASLibrary object is used to identify the DBMS schema or SAS data library, respectively, to which this library member belongs. The cardinality on this association enables a library member to belong to one DBMS schema or SAS data library.

Property

represents a SAS option. A Property object can be associated with a SASLibrary object for LIBNAME options and a SASClientConnection object for server connection options. These options will be the default set of options that are included in the LIBNAME statement for the underlying engine. A Property object can also be associated with a PhysicalTable object for data set options. These options will be the default set options that are applied when the associated library member is referenced. A Property object can also be associated with a PropertySet object. In this case, these options are applied only when the property set name is included as a value on the metadata engine LIBNAME statement options LIBOPTSET=, CONOPTSET=, or the metadata engine data set option OPTSET=. When options within a PropertySet object are used, default options are not be used.

The USERID=, PASSWORD=, and SCHEMA= options cannot be represented with a Property object. User IDs and passwords must be represented with a Login object. A DBMS schema must be represented with a DatabaseSchema object. The READ=, WRITE=, ALTER=, and PW= data set password options cannot be represented with a Property object. The SAPW= remote server access password option cannot be represented with a Property object. These passwords must be represented with a SASPassword object.

Attributes:

- *DefaultValue*—Value for the option. This attribute must be populated.
- *Delimiter*—The delimiter between the option name and the option value. For most SAS options, this is generally

an equal sign (=). This attribute will not be used if the UseValueOnly attribute is true.

- *PropertyName*—Name of the option. This attribute will not be used if the UseValueOnly attribute is true.
- *UseValueOnly*—Indicates a boolean option. When this attribute is set to true, only the DefaultValue attribute is used to specify the option. When this attribute is set to false, the PropertyName, Delimiter, and DefaultValue attributes are used to specify the option.

Associations: N/A

PropertySet

groups a set of Property objects, representing SAS options, to be used in a particular context. Property sets can be used to group a set of LIBNAME statement, connection, or data set options. The metadata engine will include a set of LIBNAME options in the LIBNAME statement for the underlying engine when a property set name is specified in the LIBOPTSET= option in the metadata engine LIBNAME statement. The owning object for these sets of Property objects is a SASLibrary object. A set of connections options are included in the LIBNAME statement for the underlying engine when a property set name is specified in the CONOPTSET= option in the metadata engine LIBNAME statement. The owning object for these sets of Property objects is a SASClientConnection object.

Here is an example of a metadata engine LIBNAME statement:

```
libname x meta repid='A5B4UB5O' libid='AB000002'
   userid=sunday pw=morning
   ipaddr='d1234.us.sas.com'
   port=9999 protocol=bridge
   liboptset=libset1
   conoptset=readset2
   ;
```

The metadata engine will apply a set of data set options when a property set name is specified in the OPTSET= data set option. The owning object for this set of Property objects is a PhysicalTable object.

```
data a;
   set x.b(optset=ds1);
run;
```

Only one set of Property objects will be used for an owning object, which is either the set of Property objects that are associated directly with the owning object or the set of Property objects that are associated with the PropertySet object specified for an owning object.

Attributes:

- *Name*—Name of the PropertySet object. This name is the value specified for the LIBOPTSET=, CONOPTSET=, and OPTSET= metadata engine options. This attribute must be populated.

Associations:

- *SetProperties*(0..n)

  This association from the PropertySet object to Property object(s) is used to obtain the options defined in the property set. The cardinality of this association enables the property set to contain many options.

SASClientConnection
represents information needed by SAS in order to connect to a server for the underlying engine.

Attributes: N/A

Associations:

- *Domain*(0..1)

  This association from the SASClientConnection object to an AuthenticationDomain object will be used to access the authorization domain that maintains the user identities (user IDs/passwords) used to access the server. The cardinality on this association enables the client connection to be associated with (at most) one authentication domain.

- *Properties*(0..n)

  This association from the SASClientConnection to Property object(s) is used to obtain the connection options to be included in the LIBNAME statement for the underlying engine. These options will be considered the default set of connection options and will be applied each time a LIBNAME statement is constructed for the underlying engine, unless a property set name is specified in the metadata engine LIBNAME statement option CONOPTSET= . The cardinality of this association enables a connection to have many connection options.

- *PropertySets*(0..n)

  This association from the SASClientConnection to PropertySet object(s) will be used to obtain a set of connection options to be included in the LIBNAME statement for the underlying engine. This association is used when a property set name is specified in the metadata engine LIBNAME statement option CONOPTSET= . If CONOPTSET= is not specified, any default connection options from the Properties association will be applied. The cardinality of this association enables a connection to have many sets of connection options.

- *SAPW*(0..1)

  This association from the SASClientConnection to a SASPassword object is used to obtain a SAS server access password option to be included in the LIBNAME statement for the underlying engine. The cardinality of this association enables a connection to have (at most) one SAS password.

SASLibrary
represents a SAS data library. A SASLibrary object is required in order to use the metadata engine. The unique identifier for the SASLibrary object must be specified by the LIBID=, LIBRARY=, or LIBURI= argument in the LIBNAME statement for the metadata engine. The metadata engine will exit with an error if the SASLibrary identifier is not included in the LIBNAME statement. Metadata included in this library will be used to construct a LIBNAME statement for the underlying engine.

Attributes:

- *Engine*—The engine that is used with this library. This attribute tells the metadata engine which engine to assign

as the underlying engine. A SAS data library will be associated with only one engine. This attribute must be populated.

- *Libref*—The libref that is used to assign the underlying engine. This attribute must be populated.
- *IsDBMSLibname*—Indicates whether this library will be used to construct a LIBNAME statement for a DBMS engine. If this attribute is true, there must be a DatabaseSchema object associated with this library ('UsingPackages' association).

Associations:

- *DefaultLogin*(0..1)

  This association from the SASLibrary object to a Login object is used to obtain the user login (user ID and password) metadata that will be used as values for the user ID and password parameters in the LIBNAME statement. However, many engines do not require user login metadata; therefore, this association might not exist. If this association does not exist, the metadata engine will attempt to obtain login metadata from the AuthenticationDomain object through the SASClientConnection object. If user login metadata cannot be obtained through the AuthenticationDomain object, the metadata engine will attempt to assign the underlying engine via a LIBNAME statement without any user login information. The cardinality of this association enables the SAS data library to use (at most) one user ID/password for the LIBNAME statement.

- *LibraryConnection*(0..1)

  This association from the SASLibrary object to a SASClientConnection object is used to obtain engine connection options. These options are represented as Property objects associated with either the SASClientConnection object or a PropertySet object that is associated with the SASClientConnection object. This association will also be used to retrieve user login metadata, where applicable, through the SASClientConnection and AuthenticationDomain objects. The cardinality on this association enables the SAS data library access to (at most) one SASClientConnection object.

- *Properties*(0..n)

  This association from the SASLibrary object to Property object(s) is used to obtain library options that should be included in the LIBNAME statement for the underlying engine. These options will be considered the default set of options and will be applied each time a LIBNAME statement is constructed for the underlying engine unless a property set name is specified in the metadata engine LIBNAME option LIBOPTSET= . The cardinality on this association enables a SAS data library to have many library options.

- *PropertySets*(0..n)

  This association from the SASLibrary object to PropertySet object(s) will be used to obtain a set of

LIBNAME options to be included in the LIBNAME statement for the underlying engine. This association is used when a property set name is specified in the metadata engine LIBNAME option LIBOPTSET= . If no LIBNAME option is specified, any default LIBNAME options from the Properties association will be applied. The cardinality of this association enables a SAS data library to have many sets of options.

- *UsingPackages*(0..1)

  This association from the SASLibrary object to either a DatabaseSchema or Directory is used to obtain metadata concerning a DBMS schema or a system directory, respectively. There should be only one DatabaseSchema or Directory associated with a SASLibrary object.

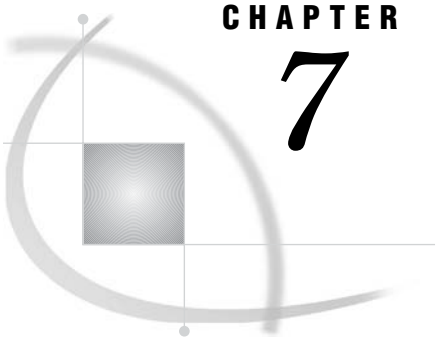
SASPassword

represents a SAS password. The READ=, WRITE=, ALTER=, and PW= data set password options cannot be represented with a Property object. The SAPW= remote engine server password option cannot be represented with a Property object. These options must be represented with a SASPassword object. SAS passwords are retrieved when the underlying engine for the metadata engine is a Base SAS engine and a password-protected SAS data set is opened . A SASPassword can also be retrieved when the LIBNAME statement is executed for the metadata engine.

Attributes: The following attributes must be populated in the SASPassword object:

- *SASPassword*—The value for a SAS data set password option or a SAS server password.
- *Type*—The name of the SAS password option. Valid values for this attribute are READ, WRITE, ALTER, PW, and SAPW.

Associations: N/A



CHAPTER

7

Metadata Requirements for Using the SAS Open Metadata Architecture Authorization Facility to Control Data Access

What Is the SAS Open Metadata Architecture Authorization Facility?

The SAS Open Metadata Architecture Authorization Facility is a subsystem of the SAS Metadata Server that renders decisions about whether an individual user or a group can take a specific action on a computing resource. The authorization facility controls access to metadata objects on the server. It can also be used to control access to the data and other actions that can be taken on the resources that the metadata objects describe.

The authorization facility uses four categories of metadata in order to render authorization decisions:

1. Identity metadata (Person, IdentityGroup, and Login metadata types) registers users on the metadata server and enables them to be associated with resources and permissions.
2. Permission metadata identifies the actions that can be taken on a resource. Repositories that are created using SAS Management Console have metadata automatically created in them that defines ReadMetadata, WriteMetadata, Read, Write, Create, and Delete permissions.
3. Resource metadata identifies the resource (for example, a library or table) that will be controlled.
4. Access control metadata relates identity and permission metadata to resource metadata to create the actual access control.

An administrator must explicitly create identity, resource, and access control metadata. The recommended method is to use SAS Management Console. Identity metadata is created in the SAS Management Console User Manager. Resource metadata is created in the Library Manager. Access control metadata is created in the Authorization Manager. Identity metadata must exist before access control metadata can be created.

For more information, see "Understanding the SAS Open Metadata Architecture Authorization Facility" in the *SAS Metadata Server: Setup and Administration Guide* , which is available from **support.sas.com/rnd/eai/openmeta**.

Understanding Access Controls

An access control is a set of metadata that grants or denies a specific identity one or more permissions to a given resource. The identity can be an individual user or a group. The resource can be a repository or a specific metadata object in a repository. The permissions that can be assigned on a metadata resource include:

READMETADATA
: specifies whether a metadata resource is available to a user.

WRITEMETADATA
: specifies whether a user can update a metadata resource; also specifies whether a user can create or delete a metadata resource in a repository.

READ
: specifies whether the data described by the metadata resource can be read by a user.

WRITE
: specifies whether the data described by the metadata resource can be updated by a user.

CREATE
: specifies whether a user can add data to the resource described by the metadata object.

DELETE
: specifies whether a user can delete data in the resource described by the metadata object.

The metadata server enforces the ReadMetadata and WriteMetadata permissions. This set of permissions is typically reserved for the metadata administrator.

The metadata engine enforces the Read, Write, Create, and Delete permissions. The Read, Write, Create, and Delete permissions are enforced on SASLibrary and PhysicalTable metadata. In order to act on the data described by a given metadata object, a user needs ReadMetadata permission and some combination of Read plus Write, Create, and Delete permissions.

For a summary of how the metadata engine enforces the Read, Write, Create, and Delete permissions, see “How the Metadata Engine Enforces Permissions” on page 43. Tables summarize how the Read, Write, Create, and Delete permissions are enforced when permission to a given object is denied and how the metadata engine behaves when the ReadMetadata and WriteMetadata permissions enforced by the SAS Metadata Server are denied.

Associating Access Controls with a Resource

An administrator can associate access controls with a resource in three ways:

- Access control information for a given resource can be defined when other properties for the resource are defined. In this way, the access control is stored directly with the resource.
- Access control information can be stored in an access control template (ACT) that can be referenced by a number of resources. The template is stored independently of any metadata resource and updated independently of any metadata resource.
- Permissions can be assigned in the default repository ACT. The default repository ACT controls who can access the repository and applies those permissions (as default permissions) to all of the objects in the repository.

A direct control is defined in the Authorization tab of a resource's Properties window. A user-defined ACT is created in the Authorization Manager and then associated with one or more resources in the Authorization tab of each resource's Properties window. The default repository ACT is maintained in the access control template folder of the Authorization Manager.

When defining access controls on a resource, you are encouraged to specify all permissions that apply to a given identity. For example, if you want to allow John Doe to read data described by a metadata object but not to create, update, or delete data, then grant John Doe ReadMetadata and Read permission, but deny him WriteMetadata, Create, Write, and Delete permissions. If you want to allow Jane Doe full access to the data described by a metadata object, grant Jane Doe ReadMetadata, Read, Write, Create, and Delete permissions on the metadata object, but deny her WriteMetadata permission.

How the Metadata Engine Enforces Permissions

The Read, Write, Create, and Delete permissions are not available to any user until an administrator sets them in the Default Repository ACT, a user-defined ACT, or in a resource's properties. If no explicit permissions are set for a user, the user inherits the permissions from the PUBLIC group in the Default Repository ACT. The metadata engine enforces Read, Write, Create, and Delete permissions on SASLibrary and PhysicalTable metadata objects. If the authorization mode is set so that the repository is secure, the metadata engine will enforce a user's permission.

The following table summarizes how the metadata engine enforces the Read, Write, Create, and Delete permissions for these objects when the permission is denied.

Table 7.1 Resource Authorization Behavior of the Metadata Engine

Metadata Object	CREATE Permission Behavior	READ Permission Behavior	WRITE Permission Behavior	DELETE Permission Behavior
SASLibrary	The user will not be able to add tables to the library. A message is issued stating that the user is not authorized to add tables to the library and processing terminates.	Behavior is not applicable for this object.	Behavior is not applicable for this object.	The user will not be able to delete tables from the library. A message is issued stating that the user is not authorized to delete tables from the library and processing terminates.
PhysicalTable	The user will not be able to add rows to the table. A message is issued stating that the user is not authorized to add data to the table and processing terminates. The user will be able to see the contents of the table. The user will be able to read data in the table (if READ permission is GRANT.)	The user will not be able to read data in the table. A message is issued stating that the user is not authorized to read data in the table and processing terminates. The user will not be able to update data, delete data, and in some cases add data to the table (any application that reads the data after it is added). The user will be able to see the contents of the table.	The user will not be able to update rows in the table. A message is issued stating that the user is not authorized to update data in the table and processing terminates. The user will be able to see the contents of the table. The user will be able to read data in the table (if READ permission is GRANT).	The user will not be able to delete rows from the table. A message is issued stating that the user is not authorized to delete data from the table and processing terminates. The user will be able to see the contents of the table. The user will be able to read data in the table (if READ permission is GRANT).

The following table summarizes the metadata engine's behavior when a ReadMetadata or WriteMetadata permission for these objects is defined.

Table 7.2 Permissions Behavior of the Metadata Engine

Metadata Object	ReadMetadata Permission Behavior	WriteMetadata Permission Behavior
SASLibrary	The user will not be able to retrieve the SASLibrary object from the repository. The user will not be able to execute a LIBNAME statement for the metadata engine. A message states that metadata cannot be retrieved or that no metadata objects are found and processing terminates.	Behavior is not applicable for this object.
PhysicalTable	The user will not be able to retrieve the PhysicalTable object from the repository. A message states that the table does not exist and processing terminates.	Behavior is not applicable for this object.
Default ACT	The user will not be able to retrieve metadata from the repository. The user will not be able to execute a LIBNAME statement for the metadata engine. A message states that metadata cannot be retrieved or that no metadata objects are found and processing terminates.	The user will not be able to create tables in or delete tables from the repository. A message states that the user is not authorized to perform this action and processing terminates.

Frequently Asked Questions about the Authorization Facility

- *How do I secure my whole repository?*

  Default user and repository permissions pertaining to all of the objects in a repository can be set on the default repository ACT under the Authorization Manager in the SAS Management Console.

- *How do I secure an individual object in my repository?*

  Individual objects can be secured under the Authorization tab in an object's Properties window in the SAS Management Console.

- *What security does a user get if the user is not registered in the repository?*

  If a user is not registered in the repository, the user has no identity in the repository. Therefore, the user's permissions will be the permissions of the PUBLIC group.

- *How do I prevent a user from creating or deleting tables in my repository?*

  Creating and deleting objects in a repository is controlled by the user's WRITEMETADATA permission on the default repository ACT. Set the user's WRITEMETADATA permission on the default repository ACT to DENY.

- *For a user to delete or update data in a table, what permissions must be granted on the table?*

  For a user to delete data from a table, the user's READ and DELETE permissions on the table must be GRANT. For a user to update data in a table, the user's READ and WRITE permission on the table must be GRANT.

- *Can a user read the data in a table and not have permission to update the data in the table?*

Yes. The user's READ permission on the table can be GRANT while the user's WRITE permission on the table is DENY.

- *How do I allow a user to add data to a table?*

  To add data to a table, the user's READ and CREATE permissions on the table must be GRANT.

- *Can a user have permissions on a library that pertain to the tables in the library?*

  Permissions are either explicitly set on a resource or inherited from a parent. If no permissions are explicitly set on a table for a user, the user's permissions for the table can be inherited from the library since the library can be a parent of a table. In this case, the user's permissions on the library will control the user's activities on the table, as well as in the library. For example, if a user's CREATE permission on a library is GRANT and the user's CREATE permission on the table is not explicitly set, the user's CREATE permission for the table can be inherited from the library and the user will be able to add data to a table as well as add tables to the library.

- *Even though the metadata engine does not enforce security on a DBMS schema, can I secure a schema instead of the individual tables in the schema?*

  Since a DBMS schema is a parent of a table, a user's permissions for a table can be inherited from the permissions on the schema. In this case, the user's permissions for each table in the schema will be the same.

- *How do I prevent a user from seeing a table in a library?*

  To hide a table in a library from a user, the user's READMETADATA permission on the table must be DENY. For the user, this table does not exist.

- *Can a user update the data in a table that the user cannot see in a library?*

  No. If a user's READMETADATA permission on the table is DENY, no metadata for the table is retrieved from the repository. When the user attempts to update the data in the table, a message is issued stating that the table does not exist.

- *Does granting a user WRITE permission on a library allow the user to update data in a table in the library?*

  No. To update data in a table in the library, the user's WRITE permission on the table, along with the READ permission, must be GRANT. The WRITE permission on the library pertains to updating the table, such as renaming the table, which is not supported by the metadata engine. Therefore, the WRITE permission on the library is not enforced by the metadata engine.

- *Can I prevent a user from viewing the contents of a library by denying the user READ permission on the library?*

  No. The READ permission on the library is not enforced. Any user, with a READMETADATA permission of GRANT on the library, can execute a LIBNAME statement for the library and view the contents of the library. To prevent the user from viewing the contents of a library, the user's READMETADATA permission on the library must be DENY.

- *If I deny a user WRITEMETADATA permission on a library, can the user add or delete tables in the library?*

  To prevent a user from adding or deleting tables in a library, the user's CREATE or DELETE permission on the library must be set to DENY.

- *If a user is denied READ permission on the whole repository but granted READ on a table, can the user read the data in the table?*

  Yes. Assuming the user's READMETADATA permission on the library and table is GRANT, the user will be able to execute a LIBNAME statement for the library and read the data in the table. The READ permission on the library is not enforced by the metadata engine.

- *How can I prevent a user from viewing the contents of a specific column on a table?*

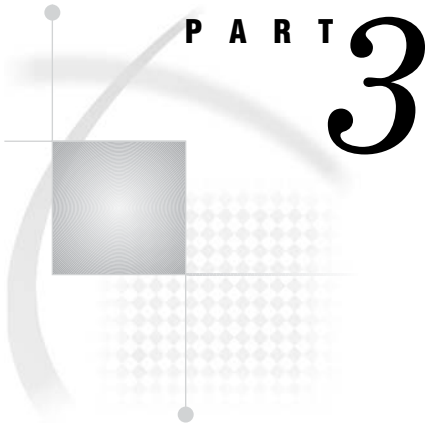
  Security on a Column object is not enforced in the metadata engine. Therefore, the metadata engine cannot prevent a user from viewing the contents of a specific column on a table.

- *How can I hide a column from a user?*

  To hide a column on a table from a user, the user's READMETADATA permission on the column must be DENY. The READMETADATA permission is enforced by the SAS Metadata Server.

- *If I secure the Directory object, does that prevent a user from accessing the library?*

  No. The metadata engine does not enforce security on a Directory object.



PART 3

Reference for the Metadata Engine

CHAPTER

8

LIBNAME Statement for the Metadata Engine

Using the LIBNAME Statement

The LIBNAME statement for the metadata engine associates a SAS libref with metadata stored on a SAS Metadata Repository in order to use the metadata engine to access data.

You must have a repository available on the SAS Metadata Server that contains metadata that defines the data to be accessed. For the necessary repository identifiers and metadata object names and identifiers, see the documentation for your application. For example, the SAS/Warehouse Administrator stores metadata for a warehouse definition. Use SAS/Warehouse Administrator as a tool in order to determine the metadata you need to identify.

The SAS Metadata Server, which is a multi-user server that stores metadata from one or more repositories, must be running in order to execute the LIBNAME statement for the metadata engine. For information about starting the server, see *SAS Metadata Server: Setup and Administration Guide*, which is available from the SAS Community **support.sas.com/rnd/eai/openmeta**.

The metadata must conform to specific metadata engine models. See Chapter 4, "SAS Metadata Model Requirements for the Metadata Engine," on page 17. For information about each metadata type, see *SAS Open Metadata Interface: Reference*.

To access the metadata, the metadata engine uses the following information that you provide on the LIBNAME statement:

IPADDR= and PORT= options
identifies the SAS Metadata Server.

REPID= or REPNAME= option
identifies the specific SAS Metadata Repository.

LIBID=, LIBURI=, or LIBRARY= argument
identifies the particular SASLibrary metadata object. The metadata engine locates the information by using the value as an anchor point into the metadata. From that starting point, the metadata engine traverses the remaining metadata, using the rules of one of the supported metadata engine models as its guide.

There are several methods that you can use in order to identify the metadata that you want to access. The following table illustrates the different modes of syntax.

Table 8.1 Specifying a Metadata Object Using the LIBNAME Statement for the Metadata Engine

Method	Example	Description
By identifier	`libname byid meta libid=A9000001 repid=A32V87R9;`	Searches for the SASLibrary object by the ID A32V87R9.A9000001.
By name	`libname byname meta library=mylib repname=myrepos;`	Searches for the SASLibrary object by resolving the name into the ID. This is the recommended method, because metadata object identifiers are not guaranteed to be constant if your repository or SASLibrary object needs to be modified.
By URI format	`libname byuri meta liburi="SASLibrary/ A32V87R9.A9000001";`	Searches for the SASLibrary object by resolving the URI format search criteria.
By identifier and name	`libname byidnam meta libid=A9000001 repname=myrepos;`	Searches for the SASLibrary object by resolving the repository name MYREPOS into the 8-character repository ID and combining it with the LIBID= identifier.
By identifier and URI format	`libname byiduri meta repid=A32V87R9 liburi="SASLibrary?@name='mylib'";`	Searches for the SASLibrary object by resolving the LIBURI= value into the Unique Identifier of the SASLibrary object that exists in the repository identified by REPID=.
By name and URI format	`libname byvaluri meta repname=myrepos liburi="SASLibrary?@name='mylib'";`	Searches for the SASLibrary object by resolving the LIBURI= value into the Unique Identifier of the SASLibrary object that exists in the repository identified by REPNAME=.

See “LIBNAME Statement Syntax” on page 53 for details on the LIBNAME statement.

LIBNAME Statement Syntax

LIBNAME *libref* META LIBID=*id* | LIBURI=*URI-format* | LIBRARY=*name*
<*connection-options*> <*engine-options*>;

Required Arguments

libref
is a valid SAS name that serves as a shortcut name to associate with metadata that is in a SAS Metadata Repository on the SAS Metadata Server. This name must conform to the rules for SAS names. A libref cannot exceed eight characters.

META
is the engine name for the metadata engine that reads metadata on the SAS Metadata Server.

LIBID=*id*
is the unique instance identifier that is assigned to a particular SASLibrary metadata object in a SAS Metadata Repository. The SASLibrary object is the anchor point from which all other metadata is obtained. The object defines the SAS data library, the engine that is used to process the data, and how it should be assigned. (The engine that is defined in the metadata is referred to as the underlying engine to the metadata engine.)

The ID can be up to 17 characters. For example, **`libid=A8000001`** or **`libid="A8000001.A8000002"`**. Using single or double quotation marks to enclose the identifier is optional.

LIBRARY | LIBRNAME=*name*
is the name that is assigned to a particular SASLibrary metadata object in a SAS Metadata Repository. The SASLibrary object is the anchor point from which all other metadata is obtained. The object defines the SAS data library, the engine that is used to process the data, and how it should be assigned. (The engine that is defined in the metadata is referred to as the underlying engine to the metadata engine.)

The name can be up to 60 characters. For example, **`library=mylib`**. Using single or double quotation marks to enclose the name is optional.

LIBURI=*URI-format*
references the particular SASLibrary metadata object in a SAS Metadata Repository using one of the SAS Open Metadata Architecture Uniform Resource Identifier (URI) formats. The SASLibrary object is the anchor point from which all other metadata is obtained. The object defines the SAS data library, the engine that is used to process the data, and how it should be assigned. (The engine that is defined in the metadata is referred to as the underlying engine to the metadata engine.)

The URI formats are as follows:

id is the unique instance identifier that is assigned to the metadata object. This format is the same as specifying LIBID=. The ID can be up to 17 characters. Using single or double quotation marks to enclose the identifier is optional.

type/id is the metadata object type and the unique instance identifier that is assigned to the metadata object. For the metadata

engine, since the object type is always SASLibrary, this format is basically the same as specifying LIBID=. Using single or double quotation marks to enclose the identifier is optional. For example, **liburi="SASLibrary/A32V87R9.A9000001"**.

type?@search-criteria
is the metadata object type and an attribute value, such as the name, engine, or libref.

Requirement: You must enclose this URI format in double quotation marks and the attribute value in single quotation marks. For example, **liburi="SASLibrary?@name='oralib'"** or **liburi="SASLibrary?@engine='base'"**.

LIBNAME Statement Options for Connecting to the SAS Metadata Server

The following LIBNAME statement options for the metadata engine establish a connection to the SAS Metadata Server and identify the metadata resources.

Note: If appropriate values are not available in order to connect to the metadata server and you execute the LIBNAME statement in the SAS windowing environment, you will be prompted for the values. △

IPADDR | HOST=*address*
is the network IP (Internet Protocol) address of the computer that hosts the SAS Metadata Server, such as **ipaddr=d6292.us.sas.com**. Using single or double quotation marks to enclose the address is optional.

The network protocol determines whether an IP address is required. If the protocol is COM and the server is on a local machine, an IP address is not required. If the protocol is COM and the server is not local (DCOM services) or the protocol is BRIDGE, an IP address is required. If this option is not specified and the protocol is specified as COM on the LIBNAME statement, this indicates a local server and no IP address will be used to connect to the server. Otherwise, if this option is not specified, the value is obtained from the METASERVER= system option. See the METASERVER= system option in *SAS Language Reference: Dictionary*.

PORT=*number*
is the TCP port that the SAS Metadata Server is listening to for connections. For example, **port=5282**. Using single or double quotation marks to enclose the number is optional.

The network protocol determines whether a port number is required. If the protocol is COM, a port number is not required. If the protocol is BRIDGE, a port number is required. If this option is not specified and the protocol is BRIDGE, the value is obtained from the METAPORT= system option or defaults to **9999**. See the METAPORT= system option in *SAS Language Reference: Dictionary*.

PROTOCOL=BRIDGE | COM
specifies the network protocol for communicating with the SAS Metadata Server. If this option is not specified, the value is obtained from the METAPROTOCOL= system option or defaults to BRIDGE. See the METAPROTOCOL= system option in *SAS Language Reference: Dictionary*.

BRIDGE
specifies that the connection use the SAS Bridge protocol. This is the default.

COM
specifies that the connection use Microsoft COM/DCOM services.

Note: When using COM services, no IP address is needed. When using DCOM services, an IP address is needed. △

PW | PASSWORD=*password*
is the password that corresponds to the user identification on the SAS Metadata Server. Using single or double quotation marks to enclose the password is optional.

The network protocol determines whether a password is required. If the protocol is COM, a password is not required; if the protocol is BRIDGE, a password is required. If this option is not specified and the protocol is BRIDGE, the value is obtained from the METAPASS= system option. See the METAPASS= system option in *SAS Language Reference: Dictionary*.

REPID=*id*
is an identifier that is assigned to a particular SAS Metadata Repository. The ID is a string that identifies the repository that stores the SASLibrary object that is specified by the LIBID= or LIBURI= argument. The ID can be up to 17 characters. Using single or double quotation marks to enclose the identifier is optional. For example, **`repid=A32V87R9`**.

Interaction: You cannot specify both REPID= and REPNAME=. If neither is specified, the value is obtained from the METAREPOSITORY= system option. See the METAREPOSITORY= system option in *SAS Language Reference: Dictionary*.

REPNAME=*name*
is a name that is assigned to a particular SAS Metadata Repository. Using single or double quotation marks to enclose the name is optional. For example, **`repname=myrepos`**.

Interaction: You cannot specify both REPNAME= and REPID=. If neither is specified, the value is obtained from the METAREPOSITORY= system option. See the METAREPOSITORY= system option in *SAS Language Reference: Dictionary*.

USERID | USER=*id*
is the user identification for logging in to the SAS Metadata Server. Using single or double quotation marks to enclose the identifier is optional.

The network protocol determines whether a user identification is required. If the protocol is COM, a user identification is not required; if the protocol is BRIDGE, a user identification is required. If this option is not specified and the protocol is BRIDGE, the value is obtained from the METAUSER= system option. See the METAUSER= system option in *SAS Language Reference: Dictionary*.

LIBNAME Statement Options for the Metadata Engine

Metadata Engine Options to Control Processing

METAOUT=ALL | META | DATA
specifies the metadata engine's processing of tables in the data source and their associated metadata in the SAS Metadata Repository.

Default: ALL

Restriction: As a LIBNAME statement option, the behavior applies to all members in the library and remains for the duration of the library. To specify METAOUT= behavior for individual tables, use the METAOUT= data set option.

ALL
specifies that you cannot create or delete tables, and you cannot create, update, or delete the associated metadata in the repository. You can create, update, and delete observations in the tables. In other words, in the terminology of input, update, and output processing for tables, you have input and update processing but not output. This is the default behavior.

Interaction: The user is restricted to only the tables that have metadata in the repository. If metadata for a table exists in the repository, any metadata authorizations are enforced for that table.

DATA
specifies that you can create, update, and delete tables but you cannot create, update, or delete the associated metadata. You can interact with the physical data source directly. All processing is supported for tables—if that processing is supported by the underlying engine that is defined in the metadata. In other words, in the terminology of input, update, and output processing for tables, you have input, update, and output.

Interaction: The user can access any table, regardless of whether it has metadata in the repository. If metadata for a table exists in the repository, any metadata authorizations are enforced for that table.

META
specifies that you can create, update, or delete metadata, but you have no access to tables. When you delete metadata for a table, METAOUT=META deletes the metadata from the repository but does not delete the table from the data source.

CAUTION:
The METAOUT=META value is not expected to be supported in the next release of the software. Use the METALIB procedure to create, update, or delete metadata. For more information about PROC METALIB, see the chapter about SAS language metadata interfaces in *SAS Open Metadata Interface: Reference*. △

Metadata Engine Options to Control Data Set Options

The following options enable you to specify which set of options to use in order to construct the LIBNAME statement for the underlying engine. In SAS Open Metadata Architecture metadata, the LIBNAME statement options for the constructed LIBNAME statement are represented using Property objects. You can use a PropertySet object to group a set of options.

If you do not specify the following options for the metadata engine, the default set of options for the constructed LIBNAME statement are used. The default set of options consist of Property objects that are directly associated to SASLibrary and SASClientConnection objects through the Properties association. For example, if you do not specify the LIBOPTSET= option, the metadata engine looks for any Property objects that are directly associated with the SASLibrary object that is specified by the LIBID= or LIBURI= argument. The metadata engine uses this default set of properties as options when constructing the LIBNAME statement for the underlying engine.

CONOPTSET='*propertyset-object*'
is a name of a PropertySet object that is associated with the SASClientConnection object that corresponds to the SASLibrary object specified by the LIBID= or LIBURI= argument. The Property objects that are associated with this

PropertySet object will be used as connection options for the constructed LIBNAME statement for the underlying engine. The name can be up to 60 characters long.

For example, in the following LIBNAME statement for the metadata engine, the PropertySet object OPTS1 represents the option CONNECTION=SHAREDREAD:

```
libname shared meta libid=a8000001 conoptset='opts1';
```

In this example, the PropertySet object OPTS2 represents the option CONNECTION=GLOBALREAD:

```
libname global meta libid=a8000001 conoptset='opts2';
```

Note: A Connection object exists only for engines that connect to a server. △

LIBOPTSET='*propertyset-object*'
is a name of a PropertySet object that is associated with the SASLibrary object specified by the LIBID= or LIBURI= argument. The Property objects that are associated with this PropertySet object will be used as statement options for the constructed LIBNAME statement for the underlying engine. The name can be up to 60 characters long.

For example, in the following LIBNAME statement for the metadata engine, the PropertySet object OPTS1 represents the options PRESERVE_COL_NAMES=YES and PRESERVE_TAB_NAMES=YES:

```
libname upcase meta libid=a8000001 liboptset='opts1';
```

In this example, the PropertySet object OPTS2 represents the options PRESERVE_COL_NAMES=NO and PRESERVE_TAB_NAMES=NO:

```
libname lowcase meta libid=a8000001 liboptset='opts2';
```

CHAPTER

9 SAS Data Set Options for the Metadata Engine

Using Data Set Options

In a SAS Metadata Repository, a PhysicalTable object represents a library member (for example, a DBMS table or a SAS data set). Property objects that are associated directly with the PhysicalTable object are used by the metadata engine as data set options. These options represent the default set of data set options that will be applied to the library member when it is processed by the underlying engine.

The following tables show how data set options are represented by individual Property objects associated with a property set. The NULLSET and IDXSET PropertySet objects are associated with a PhysicalTable object.

Table 9.1 NULLSET Data PropertySet

Data Set Option	PropertyName Attribute	Delimiter Attribute	DefaultValue Attribute	UseValueOnly Attribute
DBNULL=(EMPID=NO JOBCODE=NO)	DBNULL	=	(EMPID=NO JOBCODE=NO)	0

Table 9.2 IDXSET Data Set Property Set

Data Set Option	PropertyName Attribute	Delimiter Attribute	DefaultValue Attribute	UseValueOnly Attribute
IDXNAME=COIN_IDX	IDXNAME	=	COIN_IDX	0

METAOUT= Data Set Option

The METAOUT= data set option for the metadata engine specifies access to a table in the data source and its associated metadata in the SAS Metadata Repository.

Note: While the METAOUT= data set option enables you to specify behavior for individual tables, you can use the METAOUT= option for the LIBNAME statement in order to specify behavior for a library. However, for a library, the behavior applies to all members in the library and remains for the duration of the library. △

Note: For library procedures such as PROC DATASETS, you must specify METAOUT= as a LIBNAME statement option. You cannot specify it as a data set option. △

The syntax for the METAOUT= data set option is as follows:

METAOUT=ALL | META | DATA

Default: ALL

ALL
specifies that you cannot create or delete the table, and you cannot create, update, or delete its associated metadata. You can create, update, and delete observations in the table. In other words, in the terminology of input, update, and output processing for tables, you have input and update processing but not output. This is the default behavior of the engine.

Interaction: The user is restricted to only the tables that have metadata in the repository. If metadata exists in the repository, any metadata authorizations are enforced for the table.

DATA
specifies that you can create, update, and delete the table, but you cannot create, update, or delete its associated metadata. You can interact with the physical data source directly. All processing is supported for the table—if that processing is supported by the underlying engine that is defined in the metadata. In other words, in the terminology of input, update, and output processing for tables, you have input, update, and output.

Interaction: The user can access any table, regardless of whether it has metadata in the repository. If metadata exists in the repository, any metadata authorizations are enforced for the table.

META
specifies no access for the table, but you can create, update, or delete its associated metadata. When you delete metadata for a table, METAOUT=META deletes the metadata from the repository but does not delete the table from the data source.

CAUTION:
The METAOUT=META value is not expected to be supported in the next release of the software. Use the METALIB procedure to create, update, or delete metadata. For more information about PROC METALIB, see the chapter about SAS language metadata interfaces in *SAS Open Metadata Interface: Reference*. △

OPTSET= Data Set Option

The OPTSET= data set option for the metadata engine identifies a specific set of metadata objects in order to use specific data set options that are identified by the metadata. When the OPTSET= data set option is specified, the group of options represented by the specified PropertySet object will be applied to the library member when processed by the underlying engine. The metadata engine will retrieve the data set options from the repository so that the underlying engine can apply them.

If you do not use OPTSET=, the default set of options is used. The default set of options consist of Property objects that are directly associated with PhysicalTable objects through the Properties association.

Note: Data set options that are to be applied to referenced data must be defined in the metadata. △

The syntax for the OPTSET= data set option is as follows:

OPTSET=*propertyset-object*
is the name of a PropertySet metadata object that is associated with the PhysicalTable object that corresponds to the table that is being referenced. This metadata engine data set option is used to specify a set of data set options that will be applied to the referenced data.

Here are some examples of using the OPTSET= data set option:

- The following PRINT procedures use the Oracle SAS/ACCESS engine and the Base SAS engine:

```
proc print data=oralib.dept (dbnull=(empid=no jobcode=no));
run;


proc print data=sas9.survey (idxname=coin_idx);
run;
```

The metadata engine will retrieve the objects shown in the following table and use the corresponding metadata in the attributes.

Table 9.3 Metadata for Data Set Options

Procedure Parameter/ Option	Metadata Object	Object Attribute
dept	PhysicalTable	SASTableName='dept'
		MemberType='DATA'
dbnull	Property (associated with the dept PhysicalTable	PropertyName='dbnull'
		Delimiter='='
		DefaultValue='(empid=no jobcode=no)'
		UseValueOnly='0'
survey	PhysicalTable	SASTableName='survey'
		MemberTYpe='DATA'
idxname	Property (associated with the survey PhysicalTable)	PropertyName='idxname'
		Delimiter='='
		DefaultValue='coin_idx'
		UseValueOnly='0'

- In the following PRINT procedures, the metadata engine uses the Oracle SAS/ACCESS engine as its underlying engine. For this PROC PRINT, the default set of Property objects representing the DBNULL data set option is associated with the PhysicalTable object:

```
proc print data=x.dept;
run;
```

Then, for this PROC PRINT, the Property object representing the DBNULL data set option is associated with the NULLSET PropertySet object:

```
proc print data=x.dept (optset=nullset);
run;
```

- In the following PRINT procedures, the metadata engine uses the Base SAS engine as its underlying engine. For this PROC PRINT, the default set of Property objects representing the IDXNAME= data set option is associated with the PhysicalTable object:

```
proc print data=y.survey;
run;
```

Then, for this PROC PRINT, the Property object representing the IDXNAME= data set option is associated with the IDXSET PropertySet object:

```
proc print data=y.survey (optset=idxset);
run;
```

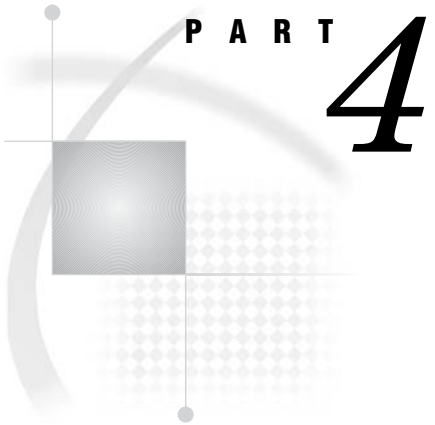
- In the following PRINT procedure, the PropertySet object OPTS1 represents the data set options READBUFF=1000 and DBLABEL=NO:

```
libname mymeta meta libid=a8000001;

proc print data=mymeta.bigtable (optset='opts1');
run;
```

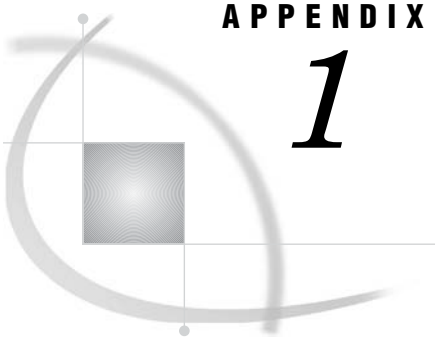
- In the following APPEND procedure, the PropertySet object OPTS2 represents the data set option DBSASTYPE=(C1='CHAR(11)'):

```
proc append base=work.sasbig data=mymeta.bigtable (optset='opts2');
run;
```



PART 4

Appendix



APPENDIX

1

Recommended Reading

Recommended Reading

Here is the recommended reading list for this title:

- *The Little SAS Book: A Primer*
- *SAS Language Reference: Concepts*
- SAS Companion that is specific to your operating environment
- Base SAS Community web site at **support.sas.com/rnd/base**
- Enterprise Integration Community web site for the SAS Open Metadata Architecture at **support.sas.com/rnd/eai/openmeta**

For a complete list of SAS publications, see the current *SAS Publishing Catalog*. To order the most current publications or to receive a free copy of the catalog, contact a SAS representative at

SAS Publishing Sales
SAS Campus Drive
Cary, NC 27513
Telephone: (800) 727-3228*
Fax: (919) 677-8166
E-mail: **sasbook@sas.com**
Web address: **support.sas.com/pubs**
* For other SAS Institute business, call (919) 677-8000.

Customers outside the United States should contact their local SAS office.

Glossary

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 file format.

libref (library reference)
a valid SAS name that serves as a shortcut name to associate with metadata objects that are in a metadata repository.

metadata
a description or definition of data or information.

metadata LIBNAME engine
the SAS engine that processes and augments data that is identified by metadata. The metadata engine retrieves information about a target SAS data library from metadata objects in a specified metadata repository.

metadata object
a set of attributes that describe a table, a server, a user, or another resource on a network. The specific attributes that a metadata object includes vary depending on which metadata model is being used.

metadata repository
a collection of related metadata objects, such as the metadata for a set of tables and columns that are maintained by an application. A SAS Metadata Repository is an example.

metadata server
a server that provides metadata management services to one or more client applications. A SAS Metadata Server is an example.

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.

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. See also SAS data set, SAS data view.

SAS data library
one or more SAS files that are accessed by the same library engine and which are referenced and stored as a unit.

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 set option
an option that appears in parentheses after a SAS data set name. Data set options specify actions that apply only to the processing of that SAS data set.

SAS data view
a type of SAS data set that retrieves data values from other files. A SAS data view contains only descriptor information such as the data types and lengths of the variables (columns), plus other information that is required for retrieving data values from other SAS data sets or from files that are stored in other software vendors' file formats. SAS data views can be created by the ACCESS and SQL procedures, as well as by the SAS DATA step.

variable
a column in a SAS data set or in a SAS data view. The data values for each variable describe a single characteristic for all observations.

Index

S

U

W

Your Turn

If you have comments or suggestions about *SAS 9.1.3 Metadata LIBNAME Engine: User's Guide, Second Edition*, please send them to us on a photocopy of this page, or send us electronic mail.

For comments about this book, please return the photocopy to

SAS Publishing
SAS Campus Drive
Cary, NC 27513
E-mail: **yourturn@sas.com**

For suggestions about the software, please return the photocopy to

SAS Institute Inc.
Technical Support Division
SAS Campus Drive
Cary, NC 27513
E-mail: **suggest@sas.com**

SAS Publishing gives you the tools to flourish in any environment with SAS®!

Whether you are new to the workforce or an experienced professional, you need a way to distinguish yourself in this rapidly changing and competitive job market. SAS Publishing provides you with a wide range of resources, from software to online training to publications to set yourself apart.

Build Your SAS Skills with SAS Learning Edition

SAS Learning Edition is your personal learning version of the world's leading business intelligence and analytic software. It provides a unique opportunity to gain hands-on experience and learn how SAS gives you the power to perform.

support.sas.com/LE

Personalize Your Training with SAS Self-Paced e-Learning

You are in complete control of your learning environment with SAS Self-Paced e-Learning! Gain immediate 24/7 access to SAS training directly from your desktop, using only a standard Web browser. If you do not have SAS installed, you can use SAS Learning Edition for all Base SAS e-learning.

support.sas.com/selfpaced

Expand Your Knowledge with Books from SAS Publishing

SAS Press offers user-friendly books for all skill levels, covering such topics as univariate and multivariate statistics, linear models, mixed models, fixed effects regression and more. View our complete catalog and get free access to the latest reference documentation by visiting us online.

support.sas.com/pubs

sas® | SAS Publishing

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.
Other brand and product names are trademarks of their respective companies. © 2005 SAS Institute Inc. All rights reserved. 345193US.0805