CHAPTER 1

Overview of the SAS/ACCESS Interface to Relational Databases

About This Document

This document provides conceptual, reference, and usage information for the SAS/ACCESS Interface to relational database management systems (DBMSs). The information in this document applies generally to all relational DBMSs that SAS/ACCESS software supports. *Because availability and behavior of SAS/ACCESS features vary from one interface to another, you should use the general information in this document with the DBMS-specific information in reference section of this document for your SAS/ACCESS interface.*

This document is intended for applications programmers and end users who meet the following conditions:

- familiar with the basics of their DBMS and its SQL (Structured Query Language)
- know how to use their operating environment
- can use basic SAS commands and statements.

Database administrators might also want to read this document to understand how the interface is implemented and administered.

Methods for Accessing Relational Database Data

SAS/ACCESS Interface to Relational Databases is a family of interfaces—each of which is licensed separately—with which you can interact with data in other vendor databases from within SAS. SAS/ACCESS provides the following methods for accessing relational DBMS data:

- You can use the LIBNAME statement to assign SAS librefs to DBMS objects such as schemas and databases. After you associate a database with a libref, you can use a SAS two-level name to specify any table or view in the database. You can then work with the table or view as you would with a SAS data set.

- You can use the Pass-Through Facility to interact with a data source using its native SQL syntax without leaving your SAS session. SQL statements are passed directly to the data source for processing.
- You can use ACCESS and DBLOAD procedures for indirect access to DBMS data. Although SAS still supports these procedures for database systems and environments on which they were available for SAS Version 6, they are no longer the recommended method for accessing DBMS data.

See "Selecting a SAS/ACCESS Method" on page 4 for information about when to use each method.

Not all SAS/ACCESS interfaces support all of these features. See "Introduction" on page 65 to determine which features are available in your environment.

Selecting a SAS/ACCESS Method

Methods for Accessing DBMS Tables and Views

In SAS/ACCESS, you can often complete a task in several ways. For example, you can access DBMS tables and views by using the LIBNAME statement or the Pass-Through Facility . Before processing complex or data-intensive operations, you might want to test several different features first to determine the most efficient one for your particular task.

SAS/ACCESS LIBNAME Statement Advantages

You should use the SAS/ACCESS LIBNAME statement for the fastest and most direct method of accessing your DBMS data except when you need to use non-ANSI-standard SQL. ANSI-standard SQL is required when you use the SAS/ACCESS library engine in the SQL procedure. However, the Pass-Through Facility accepts all SQL extensions that your DBMS provides.

Here are the advantages of using the SAS/ACCESS LIBNAME statement.

- Significantly fewer lines of SAS code are required to perform operations on your DBMS. For example, a single LIBNAME statement establishes a connection to your DBMS, lets you specify how data is processed, and lets you easily view your DBMS tables in SAS.
- You do not need to know the SQL language of your DBMS to access and manipulate data on your DBMS. You can use SAS procedures such as PROC SQL or DATA step programming on any libref that references DBMS data. You can read, insert, update, delete, and append data, and you can also create and drop DBMS tables by using SAS syntax.
- The LIBNAME statement gives you more control over DBMS operations such as locking, spooling, and data type conversion through the use of LIBNAME and data set options.
- The engine can optimize processing of joins and WHERE clauses by passing them directly to the DBMS, which takes advantage of the indexing and other processing capabilities of your DBMS. For more information, see "Overview of Optimizing Your SQL Usage" on page 37.

- The engine can pass some functions directly to the DBMS for processing.

Pass-Through Facility Advantages

Here are the advantages of using the Pass-Through Facility.

- You can use Pass-Through Facility statements so the DBMS can optimize queries, particularly when you join tables. The DBMS optimizer can take advantage of indexes on DBMS columns to process a query more quickly and efficiently.
- Pass-Through Facility statements let the DBMS optimize queries when queries have summary functions (such as AVG and COUNT), GROUP BY clauses, or columns that expressions create (such as the COMPUTED function). The DBMS optimizer can use indexes on DBMS columns to process queries more rapidly.
- On some DBMSs, you can use Pass-Through Facility statements with SAS/AF applications to handle transaction processing of DBMS data. Using a SAS/AF application gives you complete control of COMMIT and ROLLBACK transactions. Pass-Through Facility statements give you better access to DBMS return codes.
- The Pass-Through Facility accepts all extensions to ANSI SQL that your DBMS provies.

SAS/ACCESS Features for Common Tasks

The following table contains a list of tasks and the features that you can use to accomplish them.

Table 1.1 SAS/ACCESS Features for Common Tasks

Task	SAS/ACCESS Features
Read DBMS tables or views	LIBNAME statement*
	Pass-Through Facility
	View descriptors**
Create DBMS objects, such as tables	LIBNAME statement*
	DBLOAD procedure
	Pass-Through Facility's EXECUTE statement
Update, delete, or insert rows into DBMS tables	LIBNAME statement*
	View descriptors**
	Pass-Through Facility's EXECUTE statement
Append data to DBMS tables	DBLOAD procedure with APPEND option
	LIBNAME statement and APPEND procedure*
	Pass-Through Facility's EXECUTE statement
List DBMS tables	LIBNAME statement and SAS Explorer window*
	LIBNAME statement and DATASETS procedure*
	LIBNAME statement and CONTENTS procedure*
	LIBNAME statement and SQL procedure dictionary tables*

Task	SAS/ACCESS Features
Delete DBMS tables or views	LIBNAME statement and SQL procedure's DROP TABLE statement*
	LIBNAME statement and DATASETS procedure's DELETE statement*
	DBLOAD procedure with SQL DROP TABLE statement
	Pass-Through Facility's EXECUTE statement

* LIBNAME statement refers to the SAS/ACCESS LIBNAME statement.
** View descriptors refer to view descriptors that are created in the ACCESS procedure.

SAS Views of DBMS Data

SAS/ACCESS enables you to create a SAS view of data that exists in a relational database management system. A *SAS data view* defines a virtual data set that is named and stored for later use. A view contains no data, but rather describes data that is stored elsewhere. There are three types of SAS data views:

- *DATA step views* are stored, compiled DATA step programs.
- *SQL views* are stored query expressions that read data values from their underlying files, which can include SAS data files, SAS/ACCESS views, DATA step views, other SQL views, or relational database data.
- *SAS/ACCESS views* (also called view descriptors) describe data that is stored in DBMS tables. This is no longer a recommended method for accessing relational DBMS data. Use the CV2VIEW procedure to convert existing view descriptors into SQL views.

You can use all types of views as inputs into DATA steps and procedures. You can specify views in queries as if they were tables. A view derives its data from the tables or views that are listed in its FROM clause. The data accessed by a view is a subset or superset of the data in its underlying table(s) or view(s).

You can use SQL views and SAS/ACCESS views to update their underlying data if the view is based on only one DBMS table or if it is based on a DBMS view that is based on only one DBMS table and if the view has no calculated fields. You cannot use DATA step views to update the underlying data; you can use them only to read the data.

Your options for creating a SAS view of DBMS data are determined by the SAS/ACCESS feature that you are using to access the DBMS data. The following table lists the recommended methods for creating SAS views.

Table 1.2 Creating SAS Views

Feature You Use to Access DBMS Data	SAS View Technology You Can Use
SAS/ACCESS LIBNAME statement	SQL view or DATA step view of the DBMS table
SQL Procedure Pass-Through Facility	SQL view with CONNECTION TO component