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 of the relational database management systems that are supported by SAS/ACCESS software. Because the availability and behavior of SAS/ACCESS features vary from one interface to another, you should use this general document in conjunction with the documentation for your SAS/ACCESS interface. There is an individual document for each supported DBMS, and those documents are sold separately.

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

The SAS/ACCESS interface to relational databases is a family of interfaces (each of which is licensed separately) that enable you to interact with data in other vendors’ databases from within SAS. SAS/ACCESS provides the following methods for accessing relational DBMS data:

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

- The Pass-Through Facility enables you to interact with a data source using its
  native SQL syntax without leaving your SAS session. The SQL statements are
  passed directly to the data source for processing.

- The ACCESS and DBLOAD procedures support indirect access to DBMS data.
  These procedures are no longer the recommended method for accessing DBMS
data, but they continue to be supported for the database systems and
environments on which they were available for SAS Version 6.

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

Note: Not all SAS/ACCESS interfaces support all of these features. See the section
about features by host to determine which features are available in your environment.

---

**Selecting a SAS/ACCESS Method**

**Methods for Accessing DBMS Tables and Views**

In SAS/ACCESS, there are often several ways to complete a task. For example, you
can access DBMS tables and views by using the LIBNAME statement or the
Pass-Through Facility. The advantages and limitations of these features are described
below. Before processing complex or data-intensive operations, you might want to test
several of these features to determine the most efficient feature for your particular task.

**SAS/ACCESS LIBNAME Statement Advantages**

It is generally recommended that you use the SAS/ACCESS LIBNAME statement to
access your DBMS data because this is usually the fastest and most direct method. An
exception to this is 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. The
Pass-Through Facility, however, accepts all the extensions to SQL that are provided by
your DBMS.

The SAS/ACCESS LIBNAME statement has the following advantages:

- 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, enables you to specify how your data is processed, and enables you to
easily view your DBMS tables in SAS.

- You do not need to know the SQL language of your DBMS in order 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, as well as create and drop DBMS
tables by using SAS syntax.

- The LIBNAME statement provides 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 the processing of joins and WHERE clauses by passing these operations directly to the DBMS. This takes advantage of your DBMS’s indexing and other processing capabilities. 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

The Pass-Through Facility has the following advantages:

- Pass-Through Facility statements enable the DBMS to 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 enable the DBMS to optimize queries when the queries have summary functions (such as AVG and COUNT), GROUP BY clauses, or columns created by expressions (such as the COMPUTED function). The DBMS optimizer can use indexes on DBMS columns to process the queries more quickly.

- On some DBMSs, you can use Pass-Through Facility statements with SAS/AF applications to handle the transaction processing of the 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 the extensions to ANSI SQL that are provided by your DBMS.

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>
<thead>
<tr>
<th>Task</th>
<th>SAS/ACCESS Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read DBMS tables or views</td>
<td>LIBNAME statement*</td>
</tr>
<tr>
<td></td>
<td>Pass-Through Facility</td>
</tr>
<tr>
<td></td>
<td>View descriptors**</td>
</tr>
<tr>
<td>Create DBMS objects, such as tables</td>
<td>LIBNAME statement*</td>
</tr>
<tr>
<td></td>
<td>DBLOAD procedure</td>
</tr>
<tr>
<td></td>
<td>Pass-Through Facility’s EXECUTE statement</td>
</tr>
<tr>
<td>Update, delete, or insert rows into DBMS tables</td>
<td>LIBNAME statement*</td>
</tr>
<tr>
<td></td>
<td>View descriptors**</td>
</tr>
<tr>
<td></td>
<td>Pass-Through Facility’s EXECUTE statement</td>
</tr>
<tr>
<td>Append data to DBMS tables</td>
<td>DBLOAD procedure with APPEND option</td>
</tr>
<tr>
<td></td>
<td>LIBNAME statement and APPEND procedure*</td>
</tr>
<tr>
<td></td>
<td>Pass-Through Facility’s EXECUTE statement</td>
</tr>
</tbody>
</table>
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).

SQL views and SAS/ACCESS views can be used to update their underlying data if the view is based on only one DBMS table or is based on a DBMS view that is based on only one DBMS table, and if the view has no calculated fields. DATA step views cannot be used to update their underlying data; they can only 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

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