

## MANAGING LARGE DATABASES USING SAS AND ORACLE

Ronald W. Matheny, Ph.D., U. S. Environmental Protection Agency, Research Triangle Park, NC

### ABSTRACT

This paper discusses the principles of database management through the use of SAS/ASSIST to develop SAS/ACCESS code to interface with ORACLE. It examines how EPA is beginning to use the SAS interface to ORACLE to invoke interactive and batch processing for data entry, editing, updating, automatic report generation, and custom report generation functions. SAS/ACCESS is used to efficiently store the data in ORACLE, using about a third of the memory that SAS databases require and the code is written mostly through SAS/ASSIST. Customized "views" are developed using PROC SQL with PROC/ACCESS and SAS/ASSIST to efficiently manage extraction of the data from the ORACLE database into SAS for a variety of purposes, specifically database management and statistical/graphical analysis.

Detailed sample programs are not presented which demonstrate the basic principles since SAS/ASSIST uses a mouse with point-and-click screens involved so potential users may implement the concepts effectively and efficiently. Some commented SAS code (including SQL and appropriate ORACLE commands) are used as examples and discussed at a level appropriate for an beginner user of the SAS language and SAS/ASSIST.

*This paper has been reviewed in accordance with the United States Environmental Protection Agency's peer and administrative review policies and approved for presentation and publication. Mention of trade names or commercial products does not constitute endorsement or recommendation for use.*

### INTRODUCTION

Large datasets, such as used in environmental monitoring studies, are characterized by data elements related to a single sample. For example, to obtain a daily average of a compound, let us say sulfur, it is necessary to state the location of the sample, the time and date, the type of sampling device used, duplicate samples (if taken), filters and materials used in collection, analysis method and protocols, calibration of instruments, duplicate analytic processing, values derived, computerization codes, time of analysis, method of averaging, handling of non-detection levels, missing values and a host of other elements. All of these elements are not needed for every sample so it is often inefficient to use a fixed-format for the data. A relational data system is much more appropriate. ORACLE is a relational data base management system (DBMS) which links data elements together through the use of tables utilizing key variables common to each table.

Database management and data analysis, on the other hand, is much more efficient in minimizing computer resources if it is in a fixed format usually using double precision arithmetic. Resources

are saved if the data already exist in this form or the steps necessary to put the data in this form are minimized. In addition, statistical routines which use a common format are much more computer efficient as the user moves from one routine to another. This is where SAS excels.

A further difficulty, which is human related, is that of translating data in a DBMS into the form necessary for statistical analysis regardless of the statistical package. In general this requires tedious writing of extensive computer code and an extensive knowledge of both the DBMS and statistical languages. SAS/ASSIST eliminates most of the tedious work through point-and-click screen options.

The purpose of this short manuscript is to examine the efficiency in storage and retrieval of data, the ease of processing statistical data, and the movement of data from one system to another using only the SAS language, particularly SAS/ASSIST. All code is written using the UNIX "C" Shell and SAS 6.10 and ORACLE7. The code was written for the UNIX platform although the same procedures may be used on any platform. The testing of the code for this paper was completed on a Data General 532 UNIX workstation utilizing SAS and ORACLE7 on a DEC ALPHA 4000 running under OSF/1. The only differences between what is being presented and other platforms are the methods used in file references on the particular operating system. What SAS/ASSIST does is writes SAS and SQL code which accesses the ORACLE data. ORACLE does not need to be on the same machine or even the same platform. A user does not need to know how to use ORACLE in order to use the SAS interface to ORACLE through SAS/ASSIST.

### A BRIEF DESCRIPTION OF ORACLE

ORACLE is a relational Data Base Management System (DBMS) which organizes and accesses data according to relationships among data items. Data are stored in tables consisting of rows and columns; the relationships among the data elements are expressed through connections of elements in these tables. There is no significance to the order of rows and columns. The intersection of each row and each column can hold one data value. A high level language called the Structured Query Language (SQL) is used to operate and manage ORACLE data and is available in SAS through PROC SQL.

SAS/ASSIST writes the SQL code that is needed to create, edit, update and manage ORACLE databases. An ORACLE database is a collection of objects which include its tables, views, triggers, and indexes. The SAS/ACCESS product is used to create ORACLE databases, tables, views, and indexes on columns within tables. This is done by the use of the CREATE command. Through

use of the GRANT command users can be given read, write privileges on ORACLE tables, views, and indexes. SAS/ASSIST produced code may be saved for future use.

The major advantage of ORACLE over SAS is the use of the ORACLE "view" which contains no data. A view is an object consisting of a SQL SELECT statement that defines retrieval of data. The main purpose of a view is to store a commonly used query. When ORACLE accesses a view in a SQL statement it accesses the table or tables upon which the view is based. Since a view does not contain data itself but derives its data from an ORACLE table or multiple ORACLE tables, other views, a non-updatable view, or any columns in the view that are derived from an expression or aggregate (set) function are possible. Updates, inserts, and deletes are not allowed to the user. It is allowed to the owner, however. If an user has been granted privileges on the table or tables from which the view derives its data; the user still cannot access the view unless privileges on the view have been given. SAS does have PROC SQL that performs similar functions for the more advanced SAS programmer.

ORACLE "indexes" are ORACLE objects that direct ORACLE to the location in storage of a particular row for a given column. These are created with the SQL statement CREATE INDEX which assigns a name and identifies the table and column (or columns) on which the index is to be created. The index is keyed on all specified columns unless the SAS key option is used to limit the number of key columns. This allows ORACLE to retrieve data very quickly.

The SAS/ACCESS product allows views and indexes to be created from SAS/ACCESS and be treated in SAS programs as a "DATA" statement. For example, the user is able to take ORACLE data described through a SAS/ACCESS view and merge them with SAS data stored in a SAS dataset simply by issuing the commands:

```
DATA EXAMPLE;
MERGE vlib.acid sas.names;
```

In this example, the dataset "vlib.acid" is an ORACLE view and is defined as such in the LIBNAME or FILENAME statement before beginning the DATA step and. "sas.names" is a SAS dataset. When used with a SAS dataset in a SET or MERGE statement a new dataset is created (either temporary or permanent as defined in the FILENAME or LIBNAME statements) or used directly from the ORACLE table through the view created by SAS/ACCESS without creating a new dataset if the data is not manipulated through a SAS DATA statement. A PROC SQL statement can accomplish the same impact of not creating a new dataset and is preferable to using a Data step. Thus, it can be seen that SAS efficiently uses ORACLE data directly, without conversion to SAS datasets if there is no need for the conversion to take place. This saves computer time, memory, and disk space. So, an efficient SAS/ORACLE system would have predefined views that can be used directly with SAS procedures. SAS also can use SQL with the same advantages. Figure 1 illustrates the organization of a

SAS data Library. SAS/ASSIST is the simplest method of accomplishing this function.

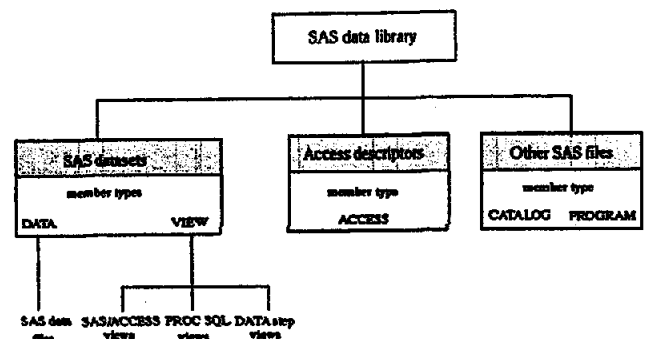


Figure 1. Organization of a SAS Data Library

## USE OF THE INTERFACE

There are several ways that the SAS/ACCESS to ORACLE interface may be used. When the ACCESS procedure is used to create an access descriptor, the procedure issues a dynamic ORACLE SQL SELECT statement to the ORACLE data dictionary tables. The ACCESS procedure then issues the equivalent of a DESCRIBE statement and one or more FETCH calls to gather information about the columns in the ORACLE table. The access descriptor's information about the ORACLE table and its columns is then copied into the view descriptor when it is created. Therefore, it is not necessary for the SAS System to call the DBMS when it creates a view descriptor. When the DBLOAD procedure is used to create an ORACLE table, the procedure issues dynamic ORACLE SQL statements to create the ORACLE table and to transfer (that is, insert) the data from a SAS data file or SAS data view into the ORACLE table. SAS/ASSIST can take over after the ORACLE database has been created.

## PROC SQL Pass-Through

Pass-through statements and the CONNECTION TO component are passed to the ORACLE server as soon as they are submitted by SAS/ASSIST. They are passed by using Pass-Through functions in the SAS System's ORACLE interface view engine. When the DATA step or any SAS procedure with a view descriptor is used, the SAS System's ORACLE interface view engine issues dynamic SQL calls to ORACLE.

The following sections explain the calls that the SAS System makes to ORACLE when the ACCESS and DBLOAD procedures or the SQL Procedure Pass-Through facility are used through SAS code or SAS/ASSIST. The connection between the SAS System and ORACLE is illustrated in Figure 2.

## Using the ACCESS Procedure to Create an Access Descriptor

The SAS/ACCESS interface view engine through SAS/ASSIST requests ORACLE to execute a SQL SELECT statement dynamically by using the ORACLE Call Interface (OCI). The following

steps are completed when the user creates an access descriptor:

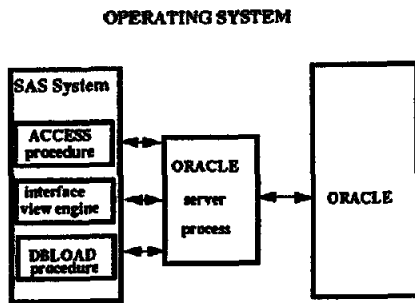


Figure 2. How SAS Interfaces with ORACLE

1. When the user supplies the username, password, and path information through SAS/ASSIST to PROC ACCESS, SAS executes the SAS/ACCESS interface, issues an OCI call to ORACLE in order to connect to the database. The connection takes place by using the client/server interface, a communication facility that provides a connection to the ORACLE server.

2. The SAS System constructs a SELECT \* FROM table-name statement through a SAS/ASSIST window and passes it to ORACLE to retrieve information about a desired table from the ORACLE data dictionary. This SELECT statement is based on the information supplied to PROC ACCESS by SAS/ASSIST and enables the SAS System to determine whether the ORACLE table exists and can be accessed.

3. The SAS/ACCESS interface issues calls to OCI to get the ORACLE table description information, such as the column names, data types (including its width, precision, and scale), and whether the columns accept null values. The results of the above steps are displayed on the ORACLE Access Descriptor Display Window through the SAS/ASSIST Window along with optional information, such as SAS variable names and formats.

#### Reading Data

When a view descriptor is used to read ORACLE data, the SAS/ACCESS interface view engine requests ORACLE to execute a SQL SELECT statement dynamically by using the ORACLE Call Interface (OCI). The interface view engine takes the following steps:

1. Using the username, password, and path information provided during creation of the access descriptor, SAS/ASSIST writes the code for the SAS/ACCESS interface which issues an OCI call to ORACLE to connect to the database. The connection takes place by using the client/server interface, a communication facility that provides a connection to the ORACLE server.

2. SAS/ASSIST constructs an ORACLE SELECT statement that is

based on the information in the view descriptor (table name, selected columns and their characteristics, and any subset of data from the Selection Criteria Entry window) and it passes this information to ORACLE.

3. The SAS System with code from SAS/ASSIST fetches the data from the ORACLE table and passes them back to the SAS procedures, as if they were observations in a SAS dataset.

4. The SAS System closes the connection with ORACLE. For example, if this SAS program executes using a view descriptor, the above steps are executed once for the PRINT procedure, then a second time for the GCHART procedure. (The data used for the two procedures are not necessarily the same because the table may have been updated by another user between procedure executions.)

#### Updating Data

A view descriptor is used to update ORACLE data in much the same way as described previously in "Reading Data." The following steps may occur:

1. Using the username, password, and path information provided during creation of the access descriptor, the SAS/ACCESS interface issues an OCI call to ORACLE to connect to the database. The connection takes place by using the client/server interface, a communication facility that provides a connection to the ORACLE server.

2. When rows are to be added to an ORACLE table, the SAS System constructs a SQL INSERT statement and passes it to ORACLE. (When a reference to a view descriptor is made in the following procedures, the ADD command in FSEDIT and FSVIEW, the APPEND procedure, or a PROC SQL INSERT statement to add data to an ORACLE table may be used.)

3. When rows are to be deleted from an ORACLE table, the SAS System constructs a SQL DELETE statement and passes it to ORACLE. (To reference a view descriptor in the following procedures, the DELETE command in FSEDIT and FSVIEW or a PROC SQLDELETE statement to delete rows from an ORACLE table are used.)

4. When data in the rows are modified, SAS/ASSIST constructs a SQL UPDATE statement and passes it to ORACLE. (To reference a view descriptor in the following procedures, the FSEDIT, the MODIFY command in FSVIEW, or a PROC SQL INSERT statement to update data in an ORACLE table are used through SAS/ASSIST). A view descriptor can also be used in the DATA step's UPDATE, MODIFY, and REPLACE statements.

#### SQL PROCEDURE PASS-THROUGH FACILITY

To read and update data with the SQL Procedure Pass-Through facility, SAS/ASSIST writes the SAS/ACCESS interface view engine code and passes ORACLE SQL statements directly to an

ORACLE database for processing. A PROC SQL CONNECT statement is passed to the SAS/ACCESS interface view engine to establish a connection with an ORACLE database, and that connection is terminated with the DISCONNECT statement. A CONNECTION TO component in a PROC SQL SELECT statement is used to read data from an ORACLE table or view. The SELECT statement (that is, PROC SQL query) can be stored as a PROC SQL view. This should be saved to be used in future SAS/ASSIST applications. A PROC SQL EXECUTE statement is used to pass any dynamic, non-query ORACLE SQL statements such as INSERT, DELETE, and UPDATE to an ORACLE database. In the EXECUTE statement and .CONNECTION TO component, all statements are passed to the ORACLE server.

## SAS/ASSIST

SAS ASSIST is a very easy way to update both SAS and ORACLE databases. Figure 3 illustrates the primary menu from SAS/ASSIST. Only one screen will be presented. Other SAS ASSIST screens are very similar. The title of the windows were changed from "ASSIST" to reflect the study which is being managed.

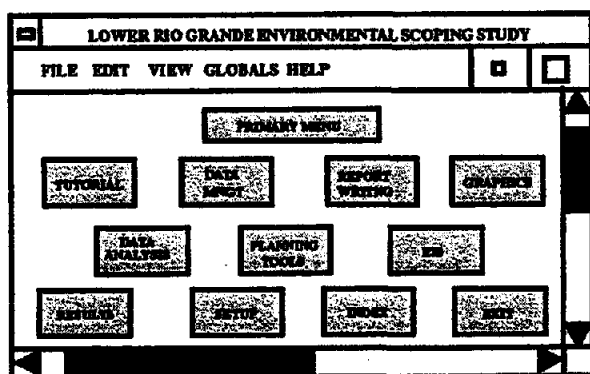


Figure 3. SAS/ASSIST Primary Menu

Probably, after the ORACLE or SAS database has been created, the system manager will wish to create a series of views for the items that need to be edited. creating a "Where Clause" and saving it as a view. This is done by using SAS/ASSIST to write SAS/ACCESS code using a PROC SQL view. or directly by clicking on the DBMS ACCESS icon on the SAS/ASSIST Data Management Menu to create the view. Multiple views are created for routine data updates and saved in a SAS LIBREF available only to the system administrator or other programmers with read/write/update privileges. The same procedure may be used with SAS databases using PROC SQL. Once a view is created, it is treated exactly as a SAS dataset would be treated, ie., Data Steps, Procedures and Macros may be used.

### The Primary Menu

When invoking the user interface the Primary Menu automatically appears on the screen. To access any feature, all the user has to do is move the cursor to the box desired, click the left button on

the mouse, and the item is automatically selected and the task is performed. Selections are described below

### Data Management Menu

Once the user enters the Data Management menu from the Primary Menu he/she is ready to select data to analyze and manipulate the data. Figure 4 illustrates the Data Management menu.

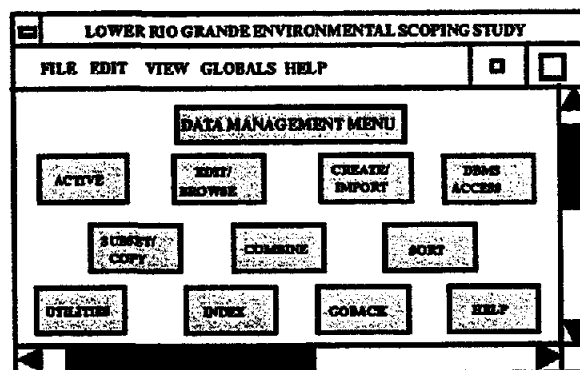


Figure 4. Data Management Window

Selecting data enables the user to choose a data set or a data view. A data view is a representation, for all practical purposes, identical to a data set. It however, contains only pointers to the master ORACLE data set so the user is always using the latest QA'd version of the data. To select a data set simply point-and-click on ACTIVE. A list of data sets will appear and the user just points-and-clicks on the data set the user desires. If the user decides not to do any other data management functions, he/she just clicks GoBack. Clicking on EDIT/BROWSE, SUBSET/COPY, COMBINE and SORT produces illustrations of what the user can do with the data and requests that the user make a choice. Variables to be analyzed are selected in an identical manner as selecting a data set through ACTIVE.

### Editing and Browsing Data

Editing the data enables the user to make changes to them, and browsing the data lets the user look at them without changing them. The user can edit/browse all of the data on the display at one time (editing and browsing data in a tabular form) or separately by each entry (editing and browsing data one record at a time). The user just selects EDIT/BROWSE from the Data Management Menu.

### Report Writing Menu

Writing reports using SAS ASSIST enables the user to produce reports of many different styles quickly. For example, the user can subset the data for the report, select the variables for the report, specify tables, subtotals, and page breaks and customize the report by adding titles, footnotes, changing page dimensions, and selecting headings. Each button selected produces an illustration

of the type of report that will be produced. Figure 5 illustrates this window.

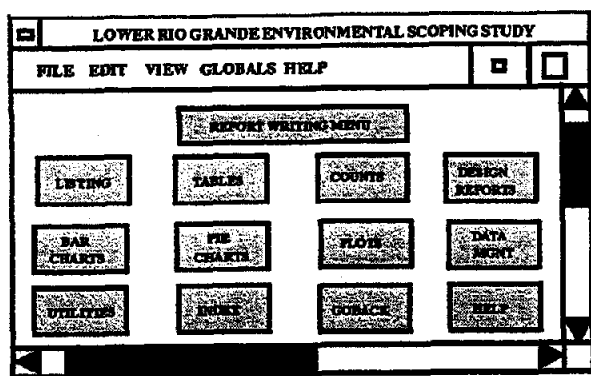


Figure 5. Report Writing Menu

The user should be aware that certain definitions apply in this interface. A **data set** is a collection of information arranged in columns and rows. An **observation**, represented by the rows in an ORACLE table is a set of data values for the same item, for example, the central site in an EPA study database for May 3, 1994 for arsenic. A **variable**, represented by the columns in a data set, is a set of data values that represents a particular type of data about all the values of arsenic for each observation.

A brief example of how to do a simple listing should illustrate well how this menu works. First, the user clicks on the Listing button, next selects an active data base if the user has not already done so through the primary window, clicks on Run and a listing of all the data in the active data set is produced. The user can move the arrow keys to look at the report. To print or save the report the user selects File from the action bar at the top of the window and from the pull-down menu selects Print or Save. The system will then ask the user the file name to save it into or if the user wants the listing sent to the default or another printer. He/she ends the Listing by just clicking on End.

SAS/ASSIST can **Subset Data** items. The user selects, by clicking on small circles, variables, constants, functions, comparison operators, logical operators, arithmetic operators and concatenations. If the user wishes, he/she may type the **WHERE** clause directly using the Edit the Where Clause button. But remember, rarely does the user have to do anything but click the mouse for any type of analysis. The exceptions are, of course, editing, adding titles, footnotes, labels, comments, and drawings. The user will probably prefer to add these things before hitting the Run button so a complete report is constructed the first time

Producing a **Tabular Report** which displays descriptive statistics in a table format is done in exactly the same way. But, the Tabular Report allows the user to place data into classes or categories for analysis and calculate statistics on these classes, such as counts, sums, means, percentages, minimums and maximums. The user

interface keeps producing menus from which the user can make choices and save or print the results in exactly the same manner in all programs.

## CONCLUSION

The user of SAS/ASSIST for Data Base Administration has just been introduced to a lot of information. Further detail can be found in the the documentation for the EPA studies, the SAS manuals, the ORACLE manuals, and any other related material. The approach used here has been an integrated approach, i.e., ORACLE is used where it is best, SAS is used where it is best, and the flexibility to add new routines written in ORACLE, SAS, FORTRAN, C, or any other compiled program. The proper "hooks" have been put in place for future use so as not to make this integrated system approach obsolete.

## REFERENCES

- Koch, G., *ORACLE7: The Complete Reference*, Osborne McGraw-Hill: New York, 1993.
- Matheny, R., "Interfacing SAS to ORACLE in the UNIX Environment", *Computing in Environmental Management*, Proceedings of the International Specialty Conference, Air & Waste Management Association, November 30 -December 2, 1994.
- Matheny, R., "Lower Rio Grande Valley Environmental Scoping Study: Data Management Guide", Draft Technical Report, United States Environmental Protection Agency, January 20, 1995.
- Matheny, R., "Managing Large Databases with Customized SAS Windows.", Accepted for presentation at the NorthEast Sas User's Group Conference, October 8 - 10, 1995.
- SAS Institute Inc., *Doing More with SAS/ASSIST Software, Version 6, First Edition*, Cary, NC: SAS Institute Inc., 1992.
- SAS Institute Inc., *Getting Started with SAS/ACCESS Software, Version 6, First Edition*, Cary, NC: SAS Institute Inc., 1994.
- SAS Institute Inc., *ORACLE7 Server SQL Language Reference Manual, First Edition*, Cary, NC: SAS Institute Inc., 1993.
- SAS Institute, Inc., *Getting Started with the SAS System Using SAS/ASSIST Software, Version 6, First Edition*, Cary, NC: SAS Institute Inc., 1993.
- SAS Institute Inc., *SAS Companion for the UNIX Environment and Derivatives, Version 6, First Edition*, Cary, NC: SAS Institute Inc., 1990.
- SAS Institute Inc., *SAS Guide to the SQL Procedure: Usage and Reference, Version 6, First Edition*, Cary, NC: SAS Institute Inc., 1989.

SAS Institute Inc., *SAS Language: Reference, Version 6, First Edition*, Cary, NC: SAS Institute Inc., 1990.

SAS Institute Inc., *SAS Technical Report P-222: Changes and Enhancements to Base SAS Software, Release 6.07*, Cary, NC: SAS Institute Inc., 1991.

SAS Institute Inc., *SAS Technical Report P-252: Changes and Enhancements to the SAS System, Release 6.09*, Cary, NC: SAS Institute Inc., 1994.

SAS Institute Inc., *SAS Technical Report P-254: Using the SQL Query Window, Release 6.08*, Cary, NC: SAS Institute Inc., 1994.

SAS Institute Inc., *SAS/ACCESS Interface to ORACLE: Usage and Reference, Version 6, Second Edition*, Cary, NC: SAS Institute Inc., 1993.

SAS Institute Inc., *SQL Language Reference Manual, Version 6.0*, Cary, NC: SAS Institute Inc., 1994.