Paper 78-28

Automatic Data File Retrieval from Different Database Engines

Xin Zhang, American College of Radiology, Philadelphia, PA
Chuanchieh Hsu, Novartis Pharmaceuticals Corporation, East Hanover, NJ

ABSTRACT

Do you need to extract data from different database engines before you analyze the data? Do you write several programs to get data from different platforms, or do you ask someone (data manager, IT personnel, etc.) to get the data you need? This paper provides a simple method allowing you to automatically extract any data files that you need for your analysis from different data engines, such as MS SQL Server, MS ACCESS, Sybase, or Oracle, using Open Database Connection (ODBC). The program is written in SAS software Version 8.01 on the Windows 2000 platform using SAS/ACCESS, SAS MACRO, and SAS/BASE.

Keywords: SAS/ACCESS, PROC SQL, ODBC, and macro.

Skill level: Beginner or intermediate level.

INTRODUCTION

Working in a clinical trial organization, you probably find your clinical trial database have been resided in many different types of database engines, such as MS Access, MS SQL Server, Sybase, or Oracle databases. In order to perform some data analyses, it is necessary to write program(s) to extract data from those different databases. The task may take a lot of time to get the data sets ready or may take a long to get someone to extract the data for you. In addition, you have to repeat the same task every time the database is updated. This task is not only inefficient but also duplicated data sets reside on different places.

The purpose of this paper is to present a method that statisticians/data analysts can extract data by themselves from MS ACCESS database, MS SQL Server 6.5 and 2000 via ODBC, as well as from Oracle database using direct connection.

EXPLANATION OF PROGRAM

Creation of a Setup Table

Before we write extraction programs, we first need to have a setup table that contains the information of the data files including filenames, locations, and database engines of the data files. The following example assumes that we are extracting five data files from four different engines. This setup table is serving like a look-up table for the program to find where each data set is located and in what format. It does not matter which database the table resides. In this example, we have a setup table in MS SQL Server 2000 to demonstrate how it works. In this database, we create a setup table, named ***file_location,*** with three columns, ***file_name, db_name and db_engine***, as follows. In the table, "FileA", in MS ACCESS format, locates in the database called "survey", "FileB1", in MS SQL Server2000, locates in the database called "clinical1", and so on.

file_name	***db_name***	***db_engine***
FileA	survey	MS ACCESS
FileB1	clinical1	SQLServer2000
FileB2	clinical1	SQLServer2000
FileC	clinical2	SQL Server 6.5
FileD	admin	Oracle
….		

The alternative way is to create such table in SAS format on your local computer, so that there is no connection to make to get those informations. For example, you can maintain a lookup table using a simple SAS program shown as below.

```
Libname out "your_path";
Data out.setup_table;
length file_name server_name db_engine $20.;
Input file_name $ server_name $ db_engine $ &;
Cards;
FileA          survey          MS ACCESS
FileB1         clinical1       SQLServer2000
FileB2         clinical1       SQLServer2000
FileC          clinical2       SQL Server 6.5
FileD          admin           Oracle
….
;;;
run;
```

Set up ODBC Connections

We open Data Sources (ODBC) window in Administrative Tools in Control Panel in Windows 2000 to set up ODBC connections.

Data Source Name	***Database Engine***
survey	MS ACCESS
clincal1	MS SQL SERVER
clinical2	MS SQL SERVER

Specify LIBNAMEs Statement to Access Databases

In SAS version 8.1, SAS/BASE provides SAS/ACCESS engine to connect to databases directly. Here, we introduce two types of connections. One is ODBC; another one is ORACLE.

ODBC Connection:
Syntax:
LIBNAME *libref <SAS/ACCESS-engine> (library-specification-1<...library-specification-n>) <options>;*

Where
Libref: A library reference name. It has still eight characters limitation, such as "survey", "main", "clinic2" and "admin".
SAS/ACCESS-engine: A database engine's name, such as ODBC, ORACLE, SYBASE.
OPTION: In ODBC connection, DSN is Data Store name, and it should be matched with the name in Control Panel ODBC setup. UID is a user login id, and PASSWORD is user's password. In ORACLE connection, there are two more options, PATH and SCHEMA.

We first need to set up the *libname* for the setup table in order to know which data file locates in what data engine.

```
LIBNAME main ODBC DSN = clinical1 UID = jeff
PASSWORD = jeff;
```

Enter File Names: We use %LET statement to enter our file names. Macro variable &file_name is a file name list separated by comma. Macro variable &totfile is a total number of files that you would like to extract.

```
%LET file_names = "FileA", "FileB1", "FileB2", "FileC",
"FileD";
%LET    totfile = 4;
```

If you have too many files to extract, and you do not want to count them, you can write the following PROC SQL procedure to get &totfile value.

```
PROC SQL noprint;
        SELECT count(file_name)
        INTO :totfile
        FROM main.file_location
        WHERE file_name in (&file_names);
QUIT;
%LET totfile = %trim(&totfile);
```

Find Files Locations

Before we extract data, we first find in which database each file was stored, and in what format the data were stored using PROC SQLwith the INTO statement to saved file names and file locations to macro variables for further usage. The information of the data file names and locations are stored into the macro variables, &file1 to &&file&tot_file, and &lctn1 to &&lctn&tot_file. The example of the SAS program is shown as below.

```
PROC SQL NOPRINT;
        SELECT file_name,  db_name
         INTO :file1 - :file&totfile , :lctn1 - :lctn&totfile
        FROM main.file_location
        WHERE file_name in (&file_names);
QUIT;

%PUT &file1 &lctn1;
%PUT &file2 &lctn2;
%PUT &file3 &lctn3;
%PUT &file4 &lctn4;
%PUT &file5 &lctn5;
```

Macro Variable Values:

&FILE1	&LCTN1:
File	survey
&FILE2	&LCTN2:
FileB1	clinical1
&FILE3	&LCTN3:
FileB2	clinical1
&FILE4	&LCTN4:
FileC	clinical2
&FILE5	&LCTN5:
FileD	admin

%GET_FILE Macro

Again, we specify the libnames and the ODBC connections first. We define the macro, %GET_FILE, to extract data, to convert, and then to save them to SAS data formats. In the FROM clause, we use a two-level names, which are database_name.table_name. For example, "survey", in here, is a *libref* name. It referred to database name. "file_data" is a table name in "survey" database.

```
LIBNAME survey ODBC DSN = survey UID = donna
PASSWORD = donna;

LIBNAME clinic2 ODBC DSN = clinical2 UID = jeff
PASSWORD = jeff;

LIBNAME admin ORACLE USER = scott PASSORD =
scott PATH = "xxx" SCHEMA = scott;

%MACRO get_file;
%DO k = 1 %TO &totfile;
%IF &&lctn&k = survey %THEN %DO;
        PROC SQL NOPRINT;
        CREATE TABLE &&file&k AS
        SELECT *
        FROM survey.access_file_data
        WHERE file_name = "&&file&k";
        QUIT;
        %END;
%IF &&lctn&k = clinical1 %THEN %DO;
        PROC SQL NOPRINT;
        CREATE TABLE &&file&k AS
        SELECT *
        FROM main.file_data
        WHERE file_name = "&&file&k";
        QUIT;
        %END;
%IF &&lctn&k = clinical2 %THEN %DO;
        PROC SQL NOPRINT;
        CREATE TABLE &&file&k AS
        SELECT *
```

```
        FROM clinc2.file
        WHERE file_name = ”&&file&k”;
        QUIT;
        %END;
%IF &&lctn&k = admin %THEN %DO;
        PROC SQL NOPRINT;
        CREATE TABLE &&file&k AS
        SELECT *
        FROM admin.file_data
        WHERE file_name = ”&&file&k”;
        QUIT;
%END;

%MEND get_file;

%get_file;
```

Disassociate libraries

Disassociate all libraries from a SAS/ACCESS library. Specify _ALL_ to list the attributes of all libraries that have *librefs* in our current session.

```
LIBNAME _ALL_ CLEAR;
```

CONCLUSIONS

We developed the program to extract data files from different database engines for statisticians/programmers to get data by themselves to perform analyses or summarize the data. Using the proposed strategy in this paper, statisticians/programmers do not need to rely on IT people to prepare the data for them, which saves a lot of time on both sides. The program is general enough to work on most of the DBMS databases, such as ORACLE, DB2, SQL Server, Microsoft Access, Sybase, and so on.

REFERENCES

SAS Institute Inc., SAS OnlineDoc®, Version 8: SAS Procedures Guide and SAS Macro Language: Reference.

ACKNOWLEDGMENTS

Special thanks go to Rex Welsh for all of his support and encouragement.

CONTACT INFORMATION

Your comments and questions are encouraged. Contact authors at:

Ms. Xin Zhang
American College of Radiology
1101 Market Street
14th Floor
Philadelphia, PA 19107
Phone: 215-574-3217
Email: xzhang@phila.acr.org

Mr. Chuanchieh Hsu
Novartis Pharmaceuticals Corporation
1 Health Plaza
East Hanover, NJ 07936
Phone: 973-781-8520
Email: jay.hsu@pharma.novartis.com