Documenting the Database: A SAS® Macro to Automate the Process
William C. Murphy
Howard M. Proskin & Associates, Inc., Rochester, NY

ABSTRACT
Creating the database for your client was straightforward, but now you have to list all of the data sets in the database, along with variable lists and sample contents. You could just run a PROC CONTENTS on all members of the library, but the client wants the variable list interleaved with sample printouts from each data set. You can easily accomplish this by alternating the PROC CONTENTS with PROC PRINT, along with appropriate TITLE statements for each data set in the library. However, to do this every time you must report on a database could be an onerous programming task. But by using ODS, PROC DATASETS, and a %DO loop, you can create a macro to handle your variable and data set documentation needs for any database.

INTRODUCTION
As a small statistical consulting firm in western New York, we build and analyze databases from data supplied by laboratory and clinical studies on a variety of consumer and medical products. The study data are usually segmented into a variety of SAS data sets. These data sets are functional groups of the study parameters and usually have relevant descriptive names (e.g. Demographics, AdverseEvents, VitalSigns, etc.). When we return this structured database to our clients, we include a list of the data sets, the variable content of each data set, and a sample printing of the first few observations from each data set. In fact, this documentation of the database is also used in-house as a guide in the analysis of the data.

MANUAL PROGRAMMING
To build this database documentation, we could simply run a PROC DATASETS to generate a list of the data sets contained in the project library. To obtain the contents of each data set, we could simply run a PROC CONTENTS on _ALL_ of the data sets in the project library. We would then run a PROC PRINT on each data set limiting the output to the first few observations.

This procedure is simple and direct and produces all of the documentation that is required (see Figure 1). However, we have to manually interleave the outputs from PROC CONTENTS and PROC PRINT. This manual collation will also mean that the page numbering will be off. Furthermore, if we use this program on a different study, we have to change the data set names in the PROC PRINTs and the number of procedures to reflect the structure of our new database.

MACRO SOLUTION
To avoid these problems, we could feed the list of data set names from PROC DATASETS into a data set. This could be readily accomplished using the SAS Output Delivery System (ODS). This new data set could be printed out to create our table-of-contents list of the complete database. We could also store the names of the data sets contained in this list into macro variables. Then using a %DO loop we could alternately run PROC CONTENTS and PROC PRINT on each data set. Thus, we would have a program in which we did not have to enter the data set names and the output would be ordered without manual intervention.

DATA SET LIST
To start out new process, we must first obtain a list of the data sets in out database and save this list into a new data set:

```sas
ods listing close;
ods output members=DataSetList;
proc datasets mt=data library=&library;
  run;
quit;
ods listing;
```

Figure 1. Documentation produced manually before collation.
The first ODS command turns off the standard output. It is similar to the NOPRINT option available in a variety of SAS PROCs. The second ODS command causes the creation of an OUTPUT data set composed of the PROC DATASETS object MEMBERS. The new data set created will be called DataSetList. We then execute a PROC DATASETS, limiting the procedure to data sets (MT=DATA) contained in the libname referred to by the macro variable &LIBRARY. The final ODS statement turns the standard output back on.

TABLE OF CONTENTS

The first page output that we create should have a list of all of the data set contained in the database. We can readily accomplish this by printing out the data set created by the PROC DATASETS:

```sas
title3 "Datasets in Library '&Library';
proc print data=DataSetList(drop=memtype)
   label nobs;
format file_size comma20.;
run;
```

where we have dropped the variable MEMTYPE, since only data sets are allowed. We also format the file size parameter to inset commas and add a descriptive title at the top of the output. The first two TITLE statements are reserved for the project description, which would be assigned by the calling program.

MACRO VARIABLES

The next thing that we need to do is save the names of the data sets contained in our database into macro variables. Using a _NULL_ DATA STEP, we can read the data set names from DataSetList and save them:

```sas
data _null_; 
   set DataSetList end=AllOver;
   call symput('DataSet'||left(_n_),memname);
   if AllOver then call symput('Tot',_n_);
run;
```

where we use the CALL SYMPUT statement to store the contents of the variable MEMNAME into macro variables. These macro variables will have the name of DataSet with a suffix supplied by the system reserved variable _n_ (i.e. &DataSet1, &DataSet2, &DataSet3…). At the end of the data set, indicated by the variable AllOver, the total number of observation (data sets) will be stored in the macro variable &tot.

LOOPING

Now we will use a macro %DO loop to produce the desired output:

```sas
%do i=1 %to &tot;
   title3 "Structure of Data Set &DataSet&i"
   proc contents
      data=&&library..&DataSet&i;
   run;
   title3 "Partial Listing of Data Set &DataSet&i"
   proc print data=&&library..&DataSet&i(obs=5) ;
   run;
%end;
```

where we again have added descriptive TITLE statements. We loop through each data set named in the macro variable &DataSet&i until we reached the last macro variable containing a data set name, indicated by &Tot. These statements will produce alternate outputs, first from PROC CONTENTS and then from PROC PRINT, of all of the data sets in our database.

DETAILS

Finally, we should clean up after ourselves. First get rid of the title so that it does not show up on the next output:

```sas
   title3;
```

Then delete the data set created and used by the macro program (DataSetList) to save memory and prevent possible conflicts in the main program:

```sas
   proc datasets nolist library=work;
   delete DataSetList;
   run;
```

We could now compile our macro and save it in a central library and everyone could have access to it (Murphy, 1998).
To invoke this macro, we must point to a library with a LIBNAME statement and then use the library name in the macro call:

```
libname base 'c:\SomeDirectory';
%Doclist(base);
```

The output generated by this macro is illustrated in Figure 2.

**CONCLUSION**

Using the SAS Output Delivery System and a simple macro program, a database composed of SAS data sets can easily be documented. Furthermore, this program can be applied to any such database and produces titled output in the desired order.

**REFERENCES**


**APPENDIX**

The following is a complete listing of the macro program that is discussed in this paper:

```
%macro DocList(library);

%***** Get List of the Data Sets in the Library *****;
ods listing close;
ods output members=DataSetList;
proc datasets mt=data library=&library;
run;
quit;
ods listing;

title3 "Datasets in Library '"&Library"';
proc print data=DataSetList(drop=memtype)
   label nobs;
   format file_size comma20.;
run;

%***** Put Data Set Names *****;
%***** into Macro Variables *****;
%***** Total Data Sets in Macro Variable *****;
data _null_;  
   set DataSetList end=AllOver;
   call symput('DataSet'||left(_n_),memname);
   if AllOver then call symput('Tot',_n_);
run;

%***** Loop through Each Data Set *****;
%do i=1 %to &tot;
   title3 "Structure of Data Set &DataSet&i";
   proc contents data=&library..&DataSet&i;
      run;
      title3 "Partial Listing of Data Set &DataSet&i";
      proc print
data=&library..&DataSet&i(obs=5) ;
      run;
%end;

%***** Clean Up *****;
   title3;
   proc datasets nolist library=work;
      delete DataSetList;
      run;
      quit;
%mend;
```

**ACKNOWLEDGEMENTS**

The author would like to express his appreciation to Howard M. Proskin for his helpful suggestions in the creation of this paper.

**TRADEMARK INFORMATION**

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.

**CONTACT INFORMATION**

Your words of wisdom are always welcomed.
Contact the author at:

William C. Murphy
Howard M. Proskin & Associates, Inc.
2468 E. Henrietta Rd.
Rochester, NY 14623
Phone 585-359-2420
FAX 585-359-0465
Web www.hmproskin.com
Email wmurphy@hmproskin.com or wcmurphy@usa.net