Programmatically Comparing Data Libraries
Stephen Hamburg, Omnicare Clinical Research, King of Prussia, PA

ABSTRACT
When two data libraries that are expected to be similar in structure contain a large number of data sets, or the data sets contain a large number of variables, a manual comparison can prove to be a difficult and tedious task. This paper demonstrates how SAS® can be used to automate the compare process by programmatically examining all of the data sets and variables that exist within two data libraries and provide a precise, summarized report of discrepancies, as well as similarities, between them.

INTRODUCTION
Often times in the pharmaceutical industry, several studies will be carried out on the same drug to analyze a variety of study populations, drug comparators, and efficacy endpoints. The databases, or data libraries, for these studies are often similar in structure, and in some instances are expected to be identical. Nevertheless, discrepancies can exist between identically named data sets, which are not always obvious. This can cause problems if you’re hoping to copy summarization programs from a prior study.

WHY NOT USE PROC COMPARE?
By now the thought of PROC COMPARE has probably crossed your mind as a possible solution. However, there are several reasons why the compare procedure would not be suitable for the aforementioned scenario.

The first thing to keep in mind is that our goal is to evaluate discrepancies with reference to the database structure – and not the data values. We should certainly expect two different studies to comprise two unique sets of data. Therefore, it is insignificant to examine the data values.

Additionally, PROC COMPARE does not provide an overall descriptive summary of the database. In order to run the PROC COMPARE, the user would be initially required to determine which data sets are common across the two data libraries. That, in particular, is part of the manual process that we are attempting to programmatically eliminate and summarize.

MY SOLUTION
As with any programming problem, there exist countless ways to attain a solution. The method which the examples below illustrates, summarizes the databases dynamically. The only user input required is the paths to the directories to be compared and, optionally, an abbreviated term to distinguish between the two directories for ease of perusing the summary output and the extension of the data set files.

Example Macro Call
```sas
%let path1=c:\nesug\final\rawdata\17Sep2006; %let path2=c:\nesug\final\rawdata\20Sep2006;
%compdb(path1=&path1, path2=&path2, prel=Old, pre2=New, ext=sas7bdat);
```

Read Data Library
The macro (COMPDB) first calls another macro (READWRIT) which reads in all data set names within each of the given data libraries, and outputs the names into one data set for each library.

```sas
%macro compdb(path1=,path2=,pre1=Old,pre2=New, ext=sas7bdat);
%macro readwrit(path=,ext=,pre=);
%local i filrf dsid rc memcount;
%* GET LIST OF ALL FILE NAMES IN THE DIRECTORY *;
libname libref "&path";
%let filrf=libref;
%let rc=%sysfunc(filename(filrf, "&path"));
%let dsid=%sysfunc(dopen(&filrf));
%let memcount=%sysfunc(dnum(&dsid));
%* WRITE OUT THE DATA SET NAMES *;
data ⪯
length setname $ 8;
%do i=1 %to &memcount;
%* Only keep files with the proper extension *;
%if %upcase(%scan(%sysfunc(dread(&dsid,&i)),-1,.))=%upcase(&ext) %then %do;
setname="%scan(%sysfunc(dread(&dsid,&i)),1,.)";
db="&pre";
output;
%end;
%end;
%let rc=%sysfunc(dclose(&dsid));
%* SORT SO CONFLICTING DATASETS ARE PRINTED FIRST *;
proc sort data=⪯ by setname;
run;
%mend readwrit;
%readwrit(path=&path1,ext=&ext,pre=&pre1);
%readwrit(path=&path2,ext=&ext,pre=&pre2);
%mend compdb;
```

Check Data Set Existence
The data sets created above are then merged together and flag variables are set, based on ‘IN’ statements, to indicate whether the data set is contained in each of the two libraries. The resulting flags are then printed along with the data set name (Sample Output 1).

```sas
data sets;
merge &PRE1(in=in1) &PRE2(in=in2);
by setname;
retain i 0;
%if in1 then &PRE1=1;
%else &PRE1=0;
%if in2 then &PRE2=1;
%else &PRE2=0;
%if in1 and in2 then do;
i+1;
call symput(compress('dsn'||put(i,3.0)),compress(setname));
call symput('N',compress(put(i,3.0)));
end;
run;
%* SORT SO CONFLICTING DATASETS ARE PRINTED FIRST *;
proc sort data=sets;
by &pre1 &pre2 setname;
run;
proc print data=sets;
var setname &PRE1 &PRE2;
title1 "List of all data set names and their library membership";
title2 "&path1.(=&pre1);
```
**Contents Procedure**

The next section of code performs a PROC CONTENTS on all of the common data sets amongst the two libraries and outputs the results to a single dataset for each of the libraries.

```sas
%* THIS MACRO LOADS THE WORK LIBRARY WITH THE COMMON *;
%* DATA SETS FROM ONE OF THE LIBRARIES OF INTEREST *;
%macro getdata(path=, out=);
%local i;
libname library "&path";
%* INITIALIZE DATASET FOR SETTING WITH CONTENTS *;
data &out;
delete;
run;
%* NOW POPULATE THE CLEAN WORK LIBRARY WITH THE DATA *;
%do i=1 %to &N;
proc contents data=library.&&dsn&i noprint out=temp;
run;
data &out;
set &out temp;
run;
%end;
%mend;
%getdata(path=&path1, out=&PRE1);
%getdata(path=&path2, out=&PRE2);
```

**Formating Data Sets**

The following data steps are intended to rename variables in order to prepare the data to be merged and also to apply labels and formats for printing purposes.

```sas
proc format;
  value typef
    1='num'
    2='char';
run;
data &PRE1(keep=memname name type length format formatd formatl label)
  rename=(type=type1 length=length1 format=form1 formatd=formatd1 formatl=formatl1 label=label1));
set &PRE1;
format type typef.;
label type="&pre1#Type" length="&pre1#Length" format="&pre1#Format" formatd="&pre1#Formatd" formatl="&pre1#Formatl" label="&pre1#Label";
run;
proc sort data=&PRE1;
  by memname name;
run;
data &PRE2(keep=memname name type length format formatd formatl label)
  rename=(type=type2 length=length2 format=form2 formatd=formatd2 formatl=formatl2 label=label2));
set &PRE2;
format type typef.;
label type="&pre2#Type" length="&pre2#Length" format="&pre2#Format" formatd="&pre2#Formatd" formatl="&pre2#Formatl" label="&pre2#Label";
run;
proc sort data=&PRE2;
  by memname name;
run;
```

**Merge Data Sets**

The last section which provides the bulk of the more detailed information, merges the two newly formatted data sets by data set name and variable name to determine differences amongst the existence of variable as well as differences amongst their attributes.

```sas
data all diff(keep=memname diffflag);
merge &PRE1(in=ok1) &PRE2(in=ok2);
by memname name;
loc='both';
if not ok1 then loc="&PRE2";
if not ok2 then loc="&PRE1";
length diffflag $3;
retain diffflag;
if first.memname then diffflag='No';
if type1^=type2 or length1^=length2 or form1^=form2 or
  formatd1^=formatd2 or formatl1^=formatl2 or
  label1^=label2 then diffflag='Yes';
diff='Yes';
diffflag='Yes';
end;
if label1^=label2 then difflbl='Yes';
else difflbl='No';
%* FIX APPEARANCE OF FORMATS FOR THE REPORT *;
  length format1 format2 $ 20;
  length len dec $ 20;
  len=put(formatl1,5.); dec=put(formatd1,5.);
  if compress(len)='0' then len=' ';
  if compress(dec)='0' then dec=' ';
  format1=compress(form1||len||'.'||dec);
  len=put(formatl2,5.); dec=put(formatd2,5.);
  if compress(len)='0' then len=' ';
  if compress(dec)='0' then dec=' ';
  format2=compress(form2||len||'.'||dec);
label memname="Data Set#Name" loc="Source#Database"
diff="Different?" difflbl="Different#Label?"
diffflag="Contain Any#Differences?";
output all;
if last.memname then output diff;
run;
```

**Print Final Results**

The findings are now printed in a summarized format (Sample Output 2), and then also in complete detail (Sample Output 3).

```sas
proc print data=diff split='#';
id memname;
var diffflag;
title1 "Common data sets with discrepancies between:";
title2 "&path1.(=&pre1);"
title3 "and";
title4 "&path2.(=&pre2);"
run;
proc print data=all split='#';
by memname name;
id memname;
var loc diffflag type1 type2 length1 length2 format1 format2
difflbl;
title1 "Comparison of Similar Data Sets from:";
```
CONCLUSION
When two data libraries are expectedly the same, the program exemplified above can be used to provide a summarized report of differences between them.

CONTRIBUTION
Your comments and questions are valued and encouraged. Contact the author at:
Stephen Hamburg
Omnicare Clinical Research
630 Allendale Road
King of Prussia, PA 19406
Stephen.Hamburg@OmnicareCR.com

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 trademarks of their respective companies.