ABSTRACT

Whether you are validating a data transfer or the data from a table or listing, using SAS® PROC COMPARE can lead to organizational nightmares for the validator due to the volume of output. This paper will present a macro that produces a condensed version of the PROC COMPARE output. The program below utilizes SASHELP.VTABLE to compare entire libraries, not just data sets.

INTRODUCTION

When using PROC COMPARE to examine the data sets of two libraries, there are several options that can be set to reduce the amount of output. However, one limitation of the standard PROC COMPARE output is that a new page begins with each new data set. Another limitation with the BRIEFSUMMARY option is that less detail is provided when a problem is encountered. This paper will step you through a program that produces PROC COMPARE results that print one observation for data sets that compare exactly, and provide all the information that a normal PROC COMPARE displays when there is a problem. The benefit of this macro is realized when a majority of the data sets compare exactly.

ASSUMPTIONS

Following are the assumptions made for this paper: All of the data sets in one library (SOURCE) are being compared to all of the data sets in a second library (COMPARE). The program uses the EXACT method of the PROC COMPARE. The programming has only been validated on the VMS platform. The program below is shown in sections as it is explained. For the program to complete, combine all of the code and submit it. This program was written in SAS v8.2, however it should work in SAS v8.1 and SAS v6.12.

DETERMINING DATA SETS TO BE COMPARED

The following code taps into the tip of the iceberg to determine the data sets that are in each library. SASHELP.VTABLE is available on most platforms and contains standard and session-specific metadata. In the example below, only the source and compare library metadata is captured. LIB1 and LIB2 macros are created to indicate the full library path for processing later. Two data sets (SOURCE contains observations from the SOURCE library; COMPARE contains observations from the COMPARE library) are created so later processing can determine that the same data sets are in each library. The last IF-THEN-ELSE statement is present in case there are no data sets in either the SOURCE or COMPARE libraries.

UNDERSTANDING THE DATA

The following data sets are in the SOURCE library: AE, BLDSAMP, CRITERIA, DEMOG, DRUGACC, END, LAB, and MHIST. The following data sets are in the COMPARE library: DEMOG, DRUGACC, END, and MHIST.

COMPARING THE LIBRARY CONTENTS

The following code verifies that the data sets in the SOURCE library are in the COMPARE library. If there are any data sets that do not compare exactly, they are output to a DISCREP data set for discrepancies. If the data sets reside in both the SOURCE and COMPARE libraries, then the data set name and number is placed in a macro variable for future processing. The number of data sets is counted and placed in the MAX macro variable for later processing as well.

COMPARING THE LIBRARY CONTENTS

The following code sorts the SOURCE and COMPARE data sets before merging them in the next data step.

The following code verifies that the data sets in the SOURCE library are in the COMPARE library. If there are any data sets that do not compare exactly, they are output to a DISCREP data set for discrepancies. If the data sets reside in both the SOURCE and COMPARE libraries, then the data set name and number is placed in a macro variable for future processing. The number of data sets is counted and placed in the MAX macro variable for later processing as well.

The following code verifies that the data sets in the SOURCE library are in the COMPARE library. If there are any data sets that do not compare exactly, they are output to a DISCREP data set for discrepancies. If the data sets reside in both the SOURCE and COMPARE libraries, then the data set name and number is placed in a macro variable for future processing. The number of data sets is counted and placed in the MAX macro variable for later processing as well.
end;
else if not a and b then do;
    comment="Data set in "||trim("&compare.")||
" but not in "||trim("&source.")||":";
    output discrep;
end;
else if a and b then do;
    if max=. then max=0;
    max=max+1;
    retain max;
    comment=" ";
    call symput(compress('dsn'||put(max,best.)),
        compress(dsn));
end;
if last then call symput('max',trim(left(put(max,8.))));
run;

COMPARING THE DATA SETS
The following code compares the SOURCE and COMPARE data sets that were found in each library. The output for the PROC COMPARE has been printed to an output file for later processing.

proc printto print=differ;
data compare;
%if &max^=0 %then %do;
    %do i=1 %to &max;
        proc compare base=source.&&dsn&i
            compare=compare.&&dsn&i
            out=&&dsn&i;
        run;
    %end;
%end;
run;
proc printto;

MASSAGING THE PROC COMPARE RESULTS
The following code reads in the output file from the PROC COMPARE and massages the data so it can be output into a "pretty" format.

After reading in the data, the IF-THEN-ELSE statement will categorize the PROC COMPARE output into two data sets: those that compare exactly (COMARES) and those that do not compare exactly (DIFFERS). For data sets to compare exactly, not only must the data match with no unequal values, but the number of variables and observations must match as well. The first IF statement reads in the name of the data set that was compared. The data set name is retained until a new data set is encountered in the PROC COMPARE output. Before moving to the next part of the IF-THEN-ELSE statement, the observation is output to the DIFFERS data set in case the libraries for that data set don't compare exactly.

data differs (where=(dsn^=' ')
    drop=len1 len2 len3 comments)
    compares (drop=len1 len2 len3 comments);
    length dsn $32.;
    infile "[differ.out] missover length=len;
    input @;
    input comment $varying132. len;
    if compress(comment)='ComparisonofSOURCE' then do;
        comments=trim(left(comment));
        len1=index(comments,'.')+1;
        len2=index(comments,'with')-2;
        len3=len2-len1+1;
        dsn=substr(comments,len1,len3);
        retain dsn;
        output differs;
    end;
else if index(comment,'differing')>0 then output differs;
    if compress(comment)="NOTE:Nounequal" then do;
        comment="No Discrepancies Found."
        output compares;
    else if compress(comment)="TheSASSystem" then delete;
    else if compress(comment)="TheCOMPAREProcedure" then delete;
    else output differs;
run;

Some of the data output to the DIFFERS data set are data sets that do compare exactly. However, the following code resolves this issue by removing data from the DIFFERS data set if it is found in the COMARES data set.

data differs;
    merge differs(in=a) compares(in=b);
    by dsn;
run;

MERGING ALL OUTPUT
The following code merges the DISCREP and COMARES data sets with the SOURCE data set to capture the number of variables and observations for the COMARES data sets.

data compares(where=(comment^=' '));
    merge compares source discrep;
    by dsn;
run;

THE OUTPUT
The following pages contain the output from the macro using the example data described earlier. Note how the information for each data set uses only one line per page when the data sets compare exactly or when the data sets are missing in one of the libraries. All other data sets that do not compare exactly are displayed as they would in a standard PROC COMPARE. In other words, the output from this macro uses 3 pages. The output from a standard PROC COMPARE would use 11 pages! Think of all the trees you could save!
Datasets with Discrepancies

Comparison of SOURCE.DEMOG with COMPARE.DEMOG
(Method=EXACT)

Data Set Summary

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Created</th>
<th>Modified</th>
<th>NVar</th>
<th>NObs</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURCE.DEMOG</td>
<td>20JAN97:08:32:48</td>
<td>20JAN97:08:32:48</td>
<td>14</td>
<td>6</td>
</tr>
<tr>
<td>COMPARE.DEMOG</td>
<td>21FEB02:15:51:19</td>
<td>21FEB02:15:51:19</td>
<td>14</td>
<td>6</td>
</tr>
</tbody>
</table>

Variables Summary

Number of Variables in Common: 14.

Observation Summary

<table>
<thead>
<tr>
<th>Observation</th>
<th>Base</th>
<th>Compare</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Obs</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>First Unequal</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Last Unequal</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Last Obs</td>
<td>6</td>
<td>6</td>
</tr>
</tbody>
</table>

Number of Observations in Common: 6.
Total Number of Observations Read from SOURCE.DEMOG: 6.
Total Number of Observations Read from COMPARE.DEMOG: 6.
Number of Observations with Some Compared Variables Unequal: 1.
Number of Observations with All Compared Variables Equal: 5.

Values Comparison Summary

Number of Variables Compared with All Observations Equal: 13.
Number of Variables Compared with Some Observations Unequal: 1.
Total Number of Values which Compare Unequal: 1.
Maximum Difference: 0.

Variables with Unequal Values

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>Len</th>
<th>Label</th>
<th>Ndif</th>
<th>MaxDif</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEX</td>
<td>CHAR</td>
<td>Sex</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Comparison of SOURCE.DEMOG with COMPARE.DEMOG
(Method=EXACT)

Value Comparison Results for Variables

<table>
<thead>
<tr>
<th>Obs</th>
<th>Base Value</th>
<th>Compare Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SEX</td>
<td>SEX</td>
</tr>
<tr>
<td></td>
<td>M</td>
<td>F</td>
</tr>
</tbody>
</table>

Datasets with Discrepancies
PRINTING THE OUTPUT
The following code prints the preceding output. As you can see, the data is output to the same file in two different formats. A condensed format is used for the data sets that compare exactly or the data sets missing in either the SOURCE or COMPARE library. A show-all format is used for the remaining data sets.

data null;
%let title1=PROC COMPARE SUMMARY;
%let title3=SOURCE Data: &lib1;
%let title4=COMPARE Data: &lib2;
len1=(132-length("&title1")/2;  
call symput("title1",repeat(" ",len1)||&title1);
len3=(132-length("&title3")/2;  
call symput("title3",repeat(" ",len3)||&title3);
len4=(132-length("&title4")/2;  
call symput("title4",repeat(" ",len4)||&title4);
len4=(132-length("&title4")/2;  
call symput("date",put(date(),date9.));  * get date;  
call symput("time",put(time(),hhmm.));  * get time;
run;

option pageno=1;
proc printto print=compare;
proc report data=compares
   spacing=0 nocenter missing headskip headline
   split="*" ls=132 ps=64;
   column ("________" dsn nvar nobs comment);
   define dsn / order width=15 spacing=0 left
   "Data set Name";
   define nvar / order width=15 spacing=4 center
   "Number of Variables";
   define nobs / order width=15 spacing=4 center
   "Number of Observations";
   define comment / order width=75 spacing=4 left
   "Comment";
   title1 "&title1";
   title3 "&title3";
   title4 "&title4";
run;

proc report data=differs
   spacing=0 nocenter missing headskip headline
   split="*" ls=132 ps=64;
   column ("________" dsn comment);
   define dsn / order order=internal noprnt;
   define comment / display width=132 spacing=0 left
   "Data sets with Discrepancies";
   break after dsn / page;
   title1 "&title1";
   title3 "&title3";
   title4 "&title4";
run;
proc printto;
%MEND;

CONCLUSION
The above program is very easy to use. In situations where most of the data sets compare exactly, it reduces the amount of paper by an average of 75% when compared to the standard PROC COMPARE. As the number of data sets increase, the amount of pages decrease! Also, if a data set was supposed to appear in a library and it doesn’t, your program doesn’t terminate!

The above program also utilizes the features provided by SASHELP.VTABLE. The program could have been written without using SASHELP.VTABLE, however, it was much more convenient and is an area that is overlooked by many. To determine if SASHELP.VTABLE is available to you, type the following and submit it. If you get output, then it is available to you.

PROC PRINT DATA=SASHELP.VTABLE;
RUN;

REFERENCES


ACKNOWLEDGMENTS
Special thanks to Jim Nezamis for opening my eyes to SASHELP.VTABLE.

Special thanks to Dan Scharf, Bonnie Duncan, and Terry Locklear for reviewing this manuscript.

CONTACT INFORMATION
Your comments and questions are valued and encouraged. Contact the author at:
Lila Thome
PPD Development, Inc.
3900-N Paramount Pkwy.
Morrisville, NC 27560
Work Phone: (919) 462-4037
Fax: (919) 654-8952
Email: lila.thome@rtp.ppdi.com