PharmaSUG 2019 - Paper BP-061

Proc Sort Revisited

Alex Chaplin, Bank of America

ABSTRACT

Proc sort can do more than sort your data. Revisit proc sort to see how you can select records and fields, rename and format fields, compress the output to save space, reuse space in your input dataset, remove and save off duplicate records and keys. Understand the difference between using the nodupkey and noduprecs options and how they affect your results at the aggregate and detail level. Software is base SAS®. Example code I've written to accompany the presentation can run in SAS University Edition or SAS On Demand for Academics because it uses SASHELP datasets as inputs. Intended audience is beginner to intermediate level SAS programmers.

INTRODUCTION

Proc sort has a number of features that can eliminate the need for prior steps to filter and format data reducing processing time, demand on system resources and storage capacity.

Using examples this papers covers the following topics related to sorting data with proc sort

- Filtering records
- Directing output
- Renaming fields
- Saving space using compress and reuse
- Eliminating duplicate keys and records
- Formatting output
- Useful system options

The following methods for sorting data are not covered in this paper but are covered in the references

- PROC SQL
- Hash tables
- Tagsort to reduce memory usage
- Collating sequences for international alphabets

Imagine you work as a SAS programmer for a Shoe Retailer. If you are able to run SAS while reading this paper take a look at the values, datatypes and formatting in SASHELP.SHOES and SASHELP.STOCKS. Understand what the values in each column represent. Now get ready to apply the capabilities of proc sort to quickly and efficiently complete each request.

BP-061.SAS INSTRUCTIONS

The code accompanying this paper, BP-061.sas, is found at the end of the paper and contains all the code examples in this paper. BP-061.sas reads from SASHELP.SHOES and SASHELP.STOCKS, which should be available under all SAS installations including SAS On Demand for Academics and SAS University Edition. You may wish to run the code as you work through the examples in this paper.

To run the code you need to change the path for libname demo.

Make the following change if running under SAS On Demand for Academics:

```
libname demo  "/home/<your SAS ODA user id>";
```

Otherwise make the following change:

```
libname demo "<your pathname>";
```

Be aware that some office applications may change straight quotes to smart quotes as you copy and paste text. You may have to change them back when you copy code to SAS to avoid a syntax error.

Alternatively you can remove all references to libname demo and create your outputs as SAS work files.

REQUEST 1 – USING THE WHERE STATEMENT WITH PROC SORT

Proc sort accepts the where statement which can often eliminate the need to filter data prior to sorting. The out statement directs sort output to a different file than the input file, thereby preserving the contents of the input file.

You have a request to extract records from the Pacific, Middle East and all European regions, or any record where sales are greater than $10,000. The output should be sorted by region in descending order and then by number of stores. The out option directs output to be in a different file than the input:

```
proc sort data=shoes_demo1(where=(Region in ('Pacific','Middle East') or
Region like('%Europe') or Sales > 10000)) out=shoes_demo2;
      by descending Region
         Stores;
run;
```

REQUEST 2 – USING DROP AND RENAME WITH PROC SORT

The Marketing department feels the product description "Slipper" is out dated and has little appeal to younger demographics. Market research indicates rebranding "Slipper" to "Indoor Footwear" will reinvigorate consumer interest leading to increased sales. Marketing plans to pilot rebranding "Slipper" to "Indoor Footwear" in the Pacific and Middle East regions

You have a request to change product name from "Slipper" to "Indoor Footwear" for the Pacific and Middle East regions.

- Use a data step to create a product2 column that has "Slipper" renamed to "Indoor Footwear"
- Use a sort step to pull records and drop / rename columns

As previously the output should be sorted by region in descending order and then by number of stores. The output should be in a different file to the input:

```
proc sort data=shoes_demo2(where=(Region in ('Pacific','Middle East'))
rename=(product2=product) drop=product) out=shoes_demo3;

      by descending Region

         Stores;

run;
```

REQUEST 3 – SAVE STORAGE USING COMPRESS AND REUSE OPTIONS

You regularly run out of storage space when running requests. This delays being able to complete requests. Although there are many ways to reduce space requirements using SAS, you want to start with something that is quick to code.

One solution is to use the system options for compress and reuse at the program level and the data level to override the program options.

- Compress provides several different compression algorithms based on what is specified after the equals option
- Reuse=yes allows SAS to access storage that has been freed up earlier in processing

Use compress=yes at the program level and override it with compress=binary at the data level:

```
options compress=yes reuse=yes;

/* Compress=binary best when mostly numeric fields */

proc sort data=demo.shoes out=shoes_demo5(compress=binary reuse=yes);

      by Region

         Product

         Subsidiary;

run;
```

Output 1 shows compression does not save space in this example. In some cases compression increases the storage space required. In these cases SAS opts by default not to apply compression. Compression rates can be over 90%.

```
NOTE: There were 395 observations read from the data set DEMO.SHOES.
NOTE: The data set WORK.SHOES_DEMO5 has 395 observations and 8
variables.
NOTE: Compressing data set WORK.SHOES_DEMO5 decreased size by 0.00
percent.
```

Output 1: Log From Proc Sort Showing Effect Of Using Compression=Binary

REQUEST 4 – ELIMINATE DUPLICATE KEYS FOR REGION AND SUBSIDIARY

A useful feature of proc sort is the ability to eliminate records where there are duplicate values in specific columns. You want to eliminate records that have duplicate keys of Region and Subsidiary, write the output and all the records with duplicate field values to separate files.

- Use the nodupkey option to eliminate records that have duplicate key values as specified in the by fields. In this case you are eliminating records that have duplicate keys of Region and Subsidiary.
- Use the out option to save the output to a file called shoes_demo6
- Use the dupout option to save the records with duplicate key values on Region and Subsidiary to a file called shoes_demo6_dups

Note how you can combine these options with other options such as the rename and drop options:

```
proc sort data=demo.shoes(rename=(product2=product) drop=product)
out=shoes_demo6 dupout=shoes_demo6_dups nodupkey;
      by Region
         Subsidiary;
run;
```

When you look at the results you will see

- 53 unique key values saved to output file
- 342 duplicate key values saved to duplicates file

REQUEST 5 – ELIMINATE DUPLICATE RECORDS WHEN SORTING BY REGION AND SUBSIDIARY

You want to eliminate duplicate records, write the output and all duplicate records to separate files.

- Use the noduprecs option to eliminate all duplicate records
- Use the out option to save the output to a file called shoes_demo7
- Use the dupout option to save the duplicate records to a file called shoes_demo7_dups

```
proc sort data=demo.shoes(rename=(product2=product) drop=product)
out=shoes_demo7 dupout=shoes_demo7_dups noduprecs;
      by Region
         Subsidiary;
run;
```

When you look at the results you will see

- All 395 input records saved to output file
- No records saved to duplicates file

This is a different result to the prior request to eliminate duplicate keys of Region and Subsidiary even though you used the same input file containing the same data in both cases.

BEWARE

- Nodupkey eliminates records with duplicate keys not duplicate records
- The exception is when using by _ALL_ with the nodupkey option which treats the entire record as a key
- If you will reference values in non-key fields after running proc sort using the nodupkey option be mindful of the effect of eliminating records with duplicate keys

Table 1 shows the file de-duped using nodupkey has lower values for aggregated amounts because more records have been eliminated as duplicates than with noduprec.

Sort Option	Region	Subsidiary	Total Store Count	Total Sales Amount	Total Return Amount
nodupkey	Africa	Addis Ababa	12	$29,761.00	$769.00
noduprecs	Africa	Addis Ababa	65	$467,429.00	$13,370.00
nodupkey	Africa	Algiers	21	$21,297.00	$710.00
noduprecs	Africa	Algiers	101	$395,600.00	$12,763.00
nodupkey	Africa	Cairo	20	$4,846.00	$229.00
noduprecs	Africa	Cairo	88	$738,198.00	$22,477.00

Table 1: The Effect of Nodupkey and Noduprecs

REQUEST 6 – USE FORMATS AND LABEL WITH PROC SORT

You get a request to show the regions as continents, show sales formatted with dollars and cents and show the best sellers in descending order along with the continent.

- Use a custom format to derive continents from regions
- Apply a dollar format to sales to show dollars and cents
- Use label and proc print to display the label for Continent

Note that we do not actually create a column for Continent. We apply a format and label to Region to appear as Continent:

```
proc format;

value $continent 'Africa'='Africa'

'Asia','Middle East','Pacific'='Asia'

'Canada','Central America/Caribbean','South America','United
States'='Americas'

'Eastern Europe','Western Europe'='Europe';

run;


proc sort data=demo.shoes out=shoes_demo8;

      format region $continent. sales dollar15.2;

      label region = 'Continent';

      by descending sales;

run;


proc print data=shoes_demo8 label;run;
```

Table 2 shows

- Region labeled as Continent
- Continent format applied to Region
- Sales formatted as dollars and cents
- Sales displayed in descending order

Continent	Product	Subsidiary	Number of Stores	Sales	Inventory	Returns
Asia	Men's Casual	Tel Aviv	11	$1,298,717.00	$2,881,005	$57,362
Americas	Men's Casual	Kingston	28	$576,112.00	$1,159,556	$20,005
Europe	Women's Casual	Copenhagen	26	$502,636.00	$1,110,412	$17,448
Africa	Men's Casual	Cairo	25	$360,209.00	$1,063,251	$9,424

Table 2: Sample Proc Print Output Showing Label And Format For Continent And Total Sales

REQUEST 7 – SELECT A DATE RANGE AND DATE FORMAT USING PROC SORT

Congratulations. You did such a great job at the Shoe Retailer that you've been hired by another company to analyze stock. You got a request to extract Microsoft stock performance between September and November 2005 ordered by date. The dates have to be in YYYY-MM-DD format.

- Use SAS date literals in the where statement
- Use between to select the date range
- Apply a date format to the date field

Note: Always specify date as 'ddmmmyy'd in the where statement but it appears as YYYY-MM-DD in the output. Be careful always to use SAS date literals in the where statement and not dates formatted in any other manner:

```
proc sort data=sashelp.stocks(where=(stock='Microsoft' and date between
'01sep05'd and '01dec05'd))

      out=demo.stocks;

      format date yymmdd10.;

      by date;

run;
```

Table 3 shows proc sort output containing records filtered for Microsoft between September and December 2005 with YYYY-MM-DD format applied to date.

Stock	Date	Open	High	Low	Close	Volume	AdjClose
Microsoft	2005-09-01	$27.38	$27.39	$25.12	$25.73	66,976,476	$25.47
Microsoft	2005-10-03	$25.71	$25.80	$24.25	$25.70	72,132,475	$25.44
Microsoft	2005-11-01	$25.61	$28.25	$25.61	$27.68	71,469,194	$27.48
Microsoft	2005-12-01	$27.73	$28.10	$26.10	$26.15	62,892,384	$25.96

Table 3: Records Sorted And Filtered For Microsoft For Specific Dates Showing Formatting Of Date

CONCLUSION

Proc sort has many options that can be combined to simplify processing, reduce storage requirements and reduce overhead. Becoming familiar with them and comfortable with how they are combined will help you produce code that is quicker to write, easier to maintain and simpler to run.

BP061.SAS

Run this code to work through the examples in the paper:

```
/* Change libname to point to your personal SAS data library */

/* If running under SAS ODA */

/*libname demo  "/home/<your SAS ODA user id>";*/

/* Otherwise */

/*libname demo "<your pathname>";*/


libname demo "<Enter pathname here";

/* See system options */

proc options;

run;

/*


 COMPRESS=NO       Specifies the type of compression to use for observations
in output SAS data sets.

 REUSE=NO          Specifies whether SAS reuses space when observations are
added to a compressed SAS data set.


 FULLSTIMER        Writes all available performance statistics to the SAS
log.


 NOSORTVALIDATE    SORT does not verify whether a data set is sorted
according to the variables in the BY statement.

 SORTDUP=PHYSICAL  Specifies whether PROC SORT removes duplicate variables
based on the DROP and KEEP options or on all data set

                   variables.

 SORTEQUALS        PROC SORT maintains the relative position in the output
data set for observations with identical BY-variable

                   values.

 SORTSEQ=          Specifies a language-specific collating sequence for the
SORT and SQL procedures.

 SORTSIZE=3221225472

                   Specifies the amount of memory that is available to the
SORT procedure.

 THREADS           Uses threaded processing for SAS applications that support
it.

*/


/* SAS products installed */
```

```
proc setinit;
run;

proc sql;
      drop table demo.shoes;
quit;

/* Memo from HQ.  Rename Slipper to Indoor Footwear. */
data demo.shoes;
      format product2 $15.;

      set sashelp.shoes;

      if product = 'Slipper'
      then product2 = 'Indoor Footwear';
      else product2 = product;
run;

proc sort data=demo.shoes out=shoes_demo1;
      by descending Region;
run;

/* Select records based on Region or Sales */
proc sort data=shoes_demo1(where=(Region in ('Pacific','Middle East') or
Region like('%Europe') or Sales > 10000)) out=shoes_demo2;
      by descending Region
         Stores;
run;

/*
For Pacific and Middle East rename product2 to product
and drop the original product column
*/
proc sort data=shoes_demo2(where=(Region in ('Pacific','Middle East'))
rename=(product2=product) drop=product) out=shoes_demo3;
      by descending Region
         Stores;
run;
```

```
/*
      Compress datasets and reuse space.
      Data level options override system options
*/
options compress=yes reuse=yes;

/* Determine most effective compression algorithm */
/* Compress=yes best when mostly character data fields */
options fullstimer;
proc sort data=demo.shoes out=shoes_demo4(compress=yes reuse=yes);
      by Region
         Product
         Subsidiary;
run;

/* Compress=binary best when mostly numeric fields */
proc sort data=demo.shoes out=shoes_demo5(compress=binary reuse=yes);
      by Region
         Product
         Subsidiary;
run;

options nofullstimer;

/* Nodupkey vs Noduprec */

/* No dupkey. Delete all duplicated values of Region and Subsidiary */
/* Write duplicates to output file */
proc sort data=demo.shoes(rename=(product2=product) drop=product)
                        out=shoes_demo6 dupout=shoes_demo6_dups nodupkey;
      by Region
         Subsidiary;
run;

/* No duprecs.  Delete all duplicate records */
/* Write duplicates to output file */
```

```
proc sort data=demo.shoes(rename=(product2=product) drop=product)
                        out=shoes_demo7 dupout=shoes_demo7_dups noduprecs;
      by Region Subsidiary;
run;

/* Effect of nodupkey vs noduprecs when aggregating non key fields */
proc sql;
      create table dedup_agg as
      select  'nodupkey' as sort_option   format $9.
               ,region
               ,subsidiary
               ,sum(stores) as store_ct
               ,sum(sales)  as sales_am         format dollar16.2
               ,sum(returns) as return_am format dollar16.2
   from shoes_demo6
   group by region
                  ,subsidiary
                  ,calculated sort_option
   union all
   select  'noduprecs' as sort_option     format $9.
               ,region
               ,subsidiary
               ,sum(stores) as store_ct
               ,sum(sales)  as sales_am         format dollar16.2
               ,sum(returns) as return_am format dollar16.2
   from shoes_demo7
   group by region
                  ,subsidiary
                  ,calculated sort_option
   order by region
                  ,subsidiary
                  ,calculated sort_option;
quit;

/* Format data */
proc format;
value $continent 'Africa'='Africa'
```

```
                        'Asia','Middle East','Pacific'='Asia'
                        'Canada','Central America/Caribbean','South
America','United States'='Americas'
                        'Eastern Europe','Western Europe'='Europe';
run;

proc sort data=demo.shoes out=shoes_demo8;
      format region $continent. sales dollar15.2;
      label region = 'Continent';
      by descending sales;
run;

proc print data=shoes_demo8 label;
run;

proc sort data=sashelp.stocks(where=(stock='Microsoft' and date between
'01sep05'd and '01dec05'd))
      out=demo.stocks;
      format date yymmdd10.;
      by date;
run;
```

REFERENCES

Hamilton, Jack. *The Problem with NODUPLICATES*, SUGI 25
http://www2.sas.com/proceedings/sugi25/25/po/25p221.pdf

Hughes, Troy Martin. *Sorting a Bajillion Records: Conquering Scalability in a Big Data World*, SESUG 2016 https://support.sas.com/resources/papers/proceedings16/11888-2016.pdf

KelseyBassett, Britta. *The SORT Procedure – Beyond the Basics*, SUGI 31
http://www2.sas.com/proceedings/sugi31/030-31.pdf

Morgan, Derek. *PROC SORT (then and) NOW*, MWSUG 2017
https://www.mwsug.org/proceedings/2017/SA/MWSUG-2017-SA04.pdf

SAS® Press. *Base SAS® 9.4 Procedures Guide, Seventh Edition*

https://documentation.sas.com/?docsetId=proc&docsetTarget=p1nd17xr6wof4sn19zkmid81p926.htm&docsetVersion=9.4&locale=en

SAS® OnDemand for Academics http://support.sas.com/software/products/ondemand-academics/#s1=1

Thewussen, Henri. *Do not waste too many resources to get your data in a specific sequence*, SAS Global Forum 2011 http://support.sas.com/resources/papers/proceedings11/242-2011.pdf

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Alex Chaplin
Bank of America
alex.chaplin@bankofamerica.com