

Paper 090-2011

ORDER, ORDER PLEASE: SORTING DATA USING PROC REPORT

Lisa Fine, United Biosource Corporation, Ann Arbor, MI

ABSTRACT

PROC REPORT is one of the more useful SAS® tools for generating formatted reports. One of the key expectations for customers of our reports is providing output in a sorted order. Accomplishing that sorted order is not always obvious or easy.

This paper clarifies and demonstrates various sorting options and techniques for achieving a desired sort order when using the REPORT procedure. The paper presents the difference between ORDER and ORDER= options and details how the two options should be used together. The available types of sorts, how to sort multiple columns when display order differs from sort order, and techniques for creating a sort variable, one which incorporates the use of an alias are also discussed.

INTRODUCTION

The purpose of this paper is to demonstrate various sorting options, and requirements, when using PROC REPORT. Specifically, the use of the ORDER options is discussed here. (Other sorting methods such as using the GROUP option are beyond the scope of this paper). There are two "ORDER" options in PROC REPORT that direct how PROC REPORT output is sorted. These are the **ORDER** usage, and the **ORDER=** options. It is important to understand the function of each option, and how the ORDER and ORDER= options work together, to ensure expected results.

SORTING DATA WITH PROC REPORT**HERE IS AN OVERVIEW:**

- The desired order of rows can be specified within the DEFINE statement. Here is an example of very basic syntax:


```
proc report data=vitals;
  column visit;
  define visit / order order=internal;
run;
```
- The two options, ORDER, and ORDER= are used in conjunction with each other to order the rows appropriately. **ORDER**, and **ORDER=** provide two separate pieces of information to PROC REPORT. It is recommended that the SAS user specify both options for ordering observations in a report.
- The ORDER usage option tells PROC REPORT that the variable's function is to be a sort variable.
- The ORDER= option tells PROC REPORT *how* to use the variable for sorting. There are four "types" of sorts that are further outlined in the next section.

ORDER USAGE OPTION VERSUS ORDER= OPTION**ORDER USAGE OPTION: WHAT VARIABLE(S) SHOULD BE USED FOR SORTING?**

The **ORDER** usage option identifies a variable as one to be used for sorting the report. For example, the PROC REPORT syntax below identifies the two variables VSTEST and VISIT as ORDER variables. Note that PROC REPORT processes variables in the order in which they are listed in the COLUMN statement (from left to right). The syntax below will produce a report that first orders the data by VSTEST and then by VISIT.

```
proc report data=vitals;
  column vsseq vstest visit;
  define vsseq / display;
  define vstest / order order=internal;
  define visit / order order=internal;
run;
```

ORDER= OPTION: HOW SHOULD THE VARIABLE(S) BE USED FOR SORTING?

The **ORDER=** option identifies the type of sorting scheme that should be used to order the rows. For example, the syntax below identifies the ORDER variable VISIT as one that should be sorted by its unformatted values (ORDER=INTERNAL).

```
proc report data=vitals;
  column vsseq visit;
  define vsseq / display;
  define visit / order order=internal;
run;
```

The four types of sorts that can be specified with the ORDER= option are:

- **ORDER=FORMATTED** Sorts by a variable's formatted values (DEFAULT)
- **ORDER=DATA** Sorts in the order that the variable values are encountered in the data set
- **ORDER=INTERNAL** Sorts by a variable's unformatted values
- **ORDER=FREQ** Sorts by frequency counts of the variable values

PROC REPORT'S DEFAULTS

ORDER=FORMATTED is the default sorting method for PROC REPORT. This is different than the default for the other SAS procedures, which default to ORDER=INTERNAL (sorting by unformatted values). All four of the ORDER= options sort by **ascending** value, unless otherwise specified. This leads to another difference between PROC REPORT and other procedures with regard to the ORDER=FREQ option. Other procedures sort by descending frequency as the default when ORDER=FREQ is specified. **The SAS user can add the word DESCENDING to the DEFINE statement with any of the above ORDER= options if a sort by descending values is the goal.**

EXAMPLES:

Below is a sample data set (VITALS), followed by comparisons of different ORDER= options.

VITALS DATA SET

VSSEQ	VSTEST	VISIT
1	SBP	0
1	SBP	2
1	SBP	3
2	DBP	0
2	DBP	1
2	DBP	2
2	DBP	3
3	PULSE	0
3	PULSE	1
3	PULSE	2

ORDER=DATA VS. ORDER=INTERNAL

The first PROC REPORT output comparison highlights the difference between ORDER=DATA and ORDER=INTERNAL. A common misunderstanding is that ORDER=DATA translates into "if the data set is already sorted how I want it ORDER=DATA will preserve this sort." This is not necessarily true. **With ORDER=DATA, sorting is determined by value order in the entire data set rather than on a per group basis.** For example, in the data set VITALS, the first VSTEST group (SBP) is missing a VISIT 1. Using ORDER=DATA establishes that the first three VISIT values in the data are 0, 2, and 3. Therefore PROC REPORT output for the following VSTESTs (DBP and PULSE) start with this order as well. For both DBP and PULSE, VISIT 1 is placed after VISITS 0, 2, and 3 (**FIGURE 1. SAMPLE below left**) even though *within these groups* VISIT 1 was originally after VISIT 0. If you desire a within group sort, ORDER=INTERNAL is the best way to obtain the appropriate VISIT sort in this case (**FIGURE 2. SAMPLE below right**).

PROC REPORT CODE AND OUTPUT**FIGURE 1. CODE ORDER=DATA**

```
proc report data=vitals;
  column vstest visit;
  define vstest /order order=data;
  define visit /order order=data;
run;
```

FIGURE 2. CODE: ORDER=INTERNAL

```
proc report data=vitals;
  column vstest visit;
  define vstest /order order=data;
  define visit /order order=internal;
run;
```

FIGURE 1. SAMPLE: ORDER=DATA

<u>VSTEST</u>	<u>VISIT</u>
SBP	0
	2
	3
DBP	0
	2
	3
	1
PULSE	0
	2
	1

FIGURE 2. SAMPLE: ORDER=INTERNAL

<u>VSTEST</u>	<u>VISIT</u>
SBP	0
	2
	3
DBP	0
	1
	2
	3
PULSE	0
	1
	2

ORDER=INTERNAL VS. ORDER=FORMATTED

The next comparison highlights the difference between ORDER=INTERNAL and ORDER=FORMATTED. In order to demonstrate ORDER=FORMATTED the FORMAT procedure and applicable values are shown directly below (values parm. and visi.). Note that you can apply a format (e.g., in the DEFINE statement) to a variable but keep the internal (unformatted value) sort. For example, in FIGURE 3. SAMPLE, while a format is applied to VSSEQ (e.g., VSSEQ 1 displays Systolic BP rather than '1'), the sort is still by VSSEQ's unformatted values 1, 2, 3. In contrast, ORDER=FORMATTED sorts by VSSEQ's formatted values (in alphabetical order). Therefore, Diastolic BP (VSSEQ=2) is listed first, then Pulse (VSSEQ=3), then Systolic BP (VSSEQ=1) (FIGURE 4. SAMPLE).

```
proc format;
  value parm
    1='Systolic BP'
    2='Diastolic BP'
    3='Pulse' ;

  value visi
    0='Visit Pre-Treatment Week 0'
    1='Treatment Week 1'
    2='Treatment Week 2'
    3='Treatment Week 3' ;
run;
```

PROC REPORT CODE AND OUTPUT**FIGURE 3. CODE ORDER=INTERNAL**

```
proc report data=vitals;
  column vsseq visit;
  define vsseq /order format=parm.
           order=internal;
  define visit /order order=internal;
run;
```

FIGURE 4. CODE: ORDER=FORMATTED

```
proc report data=vitals;
  column vsseq visit;
  define vsseq /order format=parm.
           order=formatted;
  define visit /order order=internal;
run;
```

FIGURE 3. SAMPLE: ORDER=INTERNAL

VSSEQ	<u>VISIT</u>
Systolic BP	0
	2
	3
Diastolic BP	0
	1
	2
Pulse	0
	1
	2

FIGURE 4. SAMPLE ORDER=FORMATTED

<u>VSSEQ</u>	<u>VISIT</u>
Diastolic BP	0
	1
	2
Pulse	0
	1
	2
Systolic BP	0
	2
	3

USING ONE ORDER OPTION WITHOUT THE OTHER

As stated earlier, using one of the ORDER options without the other can produce unexpected results. It is important to use the ORDER usage option to identify the variable as one that should be used to sort the observations in the report. In addition, specifying how the data should be sorted with the ORDER= option helps to ensure achieving a desired sort order. Here are examples of what can happen if both options are not utilized.

For the purpose of this example the NEWVITAL data set is used:

```
NEWVITAL DATA SET
VSSEQ  VSTEST  VISIT
  1    SBP    0
  2    DBP    0
  2    DBP    2
  3    PULSE  1
  2    DBP    3
  1    SBP    3
  3    PULSE  0
  1    SBP    2
  3    PULSE  2
  2    DBP    1
```

FIGURE 5. CODE: ORDER= but no ORDER Usage Option

```
proc report data=newvital;
  column vsseq visit;
  define vsseq / order format=parm. order=formatted;
  define visit / format=visi. order=internal;
run;
```

Without the ORDER usage option for VISIT, PROC REPORT does not recognize VISIT as an order variable. Therefore the VISIT column was not sorted, even with the ORDER=INTERNAL specification. VISIT stays in the original within-VSSEQ group order (FIGURE 5. SAMPLE, below left column).

FIGURE 6. CODE: ORDER Usage but no ORDER= Option

```
proc report data=newvital;
  column vsseq visit;
  define vsseq / order format=parm. order=formatted;
  define visit / order format=visi.;
run;
```

Without the ORDER= option for VISIT, PROC REPORT sorted VISIT, but by the default sort ORDER=FORMATTED rather than by ORDER=INTERNAL. In other words, VISIT is sorted in alphabetical order rather than by the unformatted chronological order (below center column). For example, The Visit 0 (Pre-Treatment) visits are sorted after the Treatment visits because 'V' is after 'T.' (FIGURE 6. SAMPLE, below middle column)

FIGURE 7. CODE: VISIT with both ORDER Usage and ORDER= Options

```
proc report data=newvital;
  column vsseq visit;
  define vsseq / order format=parm. order=formatted;
  define visit / order format=visi. order=internal;
run;
```

This is the desired sort. Visits are sorted in internal order which is chronological (FIGURE 7. SAMPLE, below right column).

**FIGURE 5. SAMPLE:
ORDER= WITHOUT ORDER USAGE**

**FIGURE 6. SAMPLE:
ORDER USAGE WITHOUT ORDER=**

**FIGURE 7. SAMPLE:
ORDER AND ORDER=**

<u>VSSEQ</u>	<u>VISIT</u>	<u>VSSEQ</u>	<u>VISIT</u>	<u>VSSEQ</u>	<u>VISIT</u>
Diastolic BP	Visit Pre-Treatment Week 0	Diastolic BP	Treatment Week 1	Diastolic BP	Visit Pre-Treatment Week 0
	Treatment Week 2		Treatment Week 2		Treatment Week 1
	Treatment Week 3		Treatment Week 3		Treatment Week 2
	Treatment Week 1		Visit Pre-Treatment Week 0		Treatment Week 3
Pulse	Treatment Week 1	Pulse	Treatment Week 1	Pulse	Visit Pre-Treatment Week 0
	Visit Pre-Treatment Week 0		Treatment Week 2		Treatment Week 1
	Treatment Week 2		Visit Pre-Treatment Week 0		Treatment Week 2
Systolic BP	Visit Pre-Treatment Week 0	Systolic BP	Treatment Week 2	Systolic BP	Visit Pre-Treatment Week 0
	Treatment Week 3		Treatment Week 3		Treatment Week 2
	Treatment Week 2		Visit Pre-Treatment Week 0		Treatment Week 3

SORTING A REPORT BY MULTIPLE VARIABLES

We will now switch gears to column order because this affects row order in the following way: **PROC REPORT operates on variables in your COLUMN statement from left to right**. Therefore, the order in which you list variables in the column statement is the order in which they will display. Likewise, when you have multiple sort variables the sorts will occur from left to right.

WHAT IF YOUR DESIRED DISPLAY ORDER IS DIFFERENT THAN YOUR DESIRED SORT ORDER?

For example, using the VITALBP data set below, suppose you would like to display columns in the following order: **VSSEQ, VISIT, DATEN**. However, you would like to sort by DATEN (column 3), before you sort by VISIT (column 2).

VITALBP DATA SET

VSSEQ	VISIT	DATEN
1	1	07FEB2008
2	1	07FEB2008
3	1	07FEB2008
1	99	10FEB2008
2	99	10FEB2008
3	99	10FEB2008
1	2	18FEB2008
2	2	18FEB2008
3	2	18FEB2008

The following code (FIGURE 8. CODE) will result in your desired column order because this is the order the variables are listed in the column statement. However, the code will not result in your desired row order (FIGURE 8. SAMPLE) i.e. you will have out of order dates because the visit sort took priority. **Changing the order of your DEFINE statements will NOT change the priority of the sort variables because priority is determined by the COLUMN statement.**

FIGURE 8. CODE

```
proc report data=vitalbp;
  column vsseq visit daten;
  define vsseq / order order=internal;
  define visit / order order=internal;
  define daten / order order=internal;
run;
```

FIGURE 8. SAMPLE: VISIT Sort before DATEN Sort leads to Out of Order DATES

VSSEQ	VISIT	DATEN
1	1	07FEB2008
1	2	18FEB2008
1	99	10FEB2008
2	1	07FEB2008
2	2	18FEB2008
2	99	10FEB2008
3	1	07FEB2008
3	2	18FEB2008
3	99	10FEB2008

In order to achieve the desired output, you will need to put either a copy or an alias of your date field (for sorting but not for display) before the visit field. **A copy or an alias is required because if you have more than one DEFINE statement referring to the same variable name, the last DEFINE will overwrite previous ones.** PROC REPORT will allow you to suppress the printing of this sort field with a **NOPRINT** option.

You may already have two versions of the required date field in your existing data set. For example, if you are displaying the character version of your variable (say 'DATEC' is character) you can simply sort by the numeric version of DATE (say 'DATEN' in this example). Two additional ways to accomplish the desired sort are detailed below.

EXAMPLE 1: Create sort variables in the data set that feeds PROC REPORT. This variable is simply a copy of the variable you are displaying (FIGURE 9. CODE).

FIGURE 9. CODE

```

**create sorting variable daten2;
data vitalbp;
  set vitalbp;
  daten2=daten;
run;

**create report;
proc report data=vitalbp;
  column vsseq daten2 visit daten;
  define vsseq / order order=internal;
  define daten2 / order order=internal noprint;
  define visit / order order=internal;
  define daten / order order=internal;
run;

```

EXAMPLE 2: Create an alias in PROC REPORT. This method is utilized within the PROC and does not impact the incoming data set (FIGURE 10. CODE).

An alias is the same variable, but with an alternate name so the variable can be defined in more than one way. The alias is assigned in the column statement as such: VARIABLE=ALIAS.

FIGURE 10. CODE

```

**create sort variable with alias;
proc report data=vitalbp;
  column vsseq daten=daten2 visit daten;
  define vsseq / order order=internal;
  define daten2 / order order=internal noprint;
  define visit / order order=internal;
  define daten / order order=internal;
run;

```

Both FIGURE 9. and FIGURE 10. code will result in the correct output in which the DATE sort takes priority over the VISIT sort.

Do I want the report to be sorted by ascending or descending values? Neither!

A more challenging situation arises when we want a sort we cannot get to by using ascending or descending sorts with the existing variable values.

For example, we can easily obtain the following orders of VSSEQ:

Abbreviations

SBP, DBP, PULSE:	define vssseq / order order=internal (ascending sort of values 1, 2, 3)
PULSE, DBP, SBP:	define vssseq / order order=internal descending (descending sort of values 1, 2, 3)
DBP, PULSE, SBP:	define vssseq / order order=formatted (ascending sort of values Diastolic BP, Pulse, Systolic BP)
SBP, PULSE, DBP:	define vssseq / order order=formatted DESCENDING (descending sort of values Diastolic BP, Pulse, Systolic BP)

But what if you want to sort rows in the following order: **DBP, SBP, PULSE**? Neither the unformatted nor formatted values will sort this way with ascending or descending options.

The alias method is one way to accomplish this sort without having to go back to your data set. FIGURE 11. CODE demonstrates code for resorting the data in a non-ascending / non-descending manner. First, PROC FORMAT is used to create new sorting values that can be applied to VSSEQ. NOTE, asterisks are included next to the alphanumeric values simply for readability in this paper because the output (FIGURE 11. SAMPLE) contains two VSSEQ columns. The VSSEQ with the values 1*, 2*, 3* came from the DEFINE statement for the alias VSSEQ2. A label could have been applied to differentiate the column headers, but this example is meant to demonstrate that 'VSSEQ2' in the column and define statements is simply an alternate name for the same variable. One additional note, while you can perform different operations such as calculating various statistics and applying different formats to the variable and aliases, the usage option cannot contradict what was used for the variable's first define statement. For example, if you assign the first VSSEQ (called VSSEQ2) as an ORDER variable, you will get the following error if you try to assign VSSEQ as a DISPLAY variable in the second DEFINE statement: "DISPLAY conflicts with earlier use of VSSEQ."

FIGURE 11. CODE

```
proc format;
  value newsort
    1 = '1*' /*dbp*/
    2 = '3*' /*pulse*/
    3 = '2*' /*sbp*/
  ;
run;

**create report with alias;
proc report data=vitalbp;
  column vsseq=vsseq2 vsseq visit;
  define vsseq2 / order order=formatted format=newsort.;
  define vsseq / order order=formatted format=parm.;
  define visit / order order=internal;
run;
```

FIGURE 11. SAMPLE: VSSEQ Order is not Ascending nor Descending

VSSEQ	VSSEQ	VISIT
1*	Diastolic BP	1
		2
		99
2*	Systolic BP	1
		2
		99
3*	Pulse	1
		2
		99

CONCLUSION

The ORDER and ORDER= options provide different pieces of information to PROC REPORT. Omitting either option can lead to unexpected results. For the SAS user to achieve the greatest control over sorting of one's report the author recommends specifying **ORDER ORDER={option}** in each order variable's DEFINE statement.

REFERENCES

- Carpenter, Arthur L. (2005), PROC REPORT Basics: Getting Started with the Primary Statements, *Proceedings of the 2005 Pharmaceutical SAS User Group Conference*, Paper HOW07, Chapel Hill, NC: PharmaSUG.
- McMahill, Allison (2007), Beyond the Basics: Advanced PROC REPORT Tips and Tricks, *Proceedings of the 2007 SAS Global Forum*, Paper 276-2007, Cary, NC: SAS Institute Inc.

- Pass, Ray and Huang, Ya (2002), Re: Proc report, problem using order ?, "SAS(r) Discussion" <SAS-L@LISTSERV.UGA.EDU, Date: Thu, 25 Jul 2002 15:15:37 -0400 From: Ray Pass, Subject Comments: To: Huang, Ya, (LISTSERV at University of Georgia), Available at <http://www.listserv.uga.edu/cgi-bin/wa?A2=ind0207D&L=sas-l&P=40236>.
- Pass, Ray and Ewing, Daphne (2006), So You're Still Not Using Proc Report. Why Not?, *Proceedings of SAS Users Group International 31 Conference*, Paper 235-31, Cary, NC: SAS Institute Inc.
- Sharpe, Jason R. (2009), Ordering Data Values: Procedure Defaults and options, SAS TS Docs TS-407, Cary, NC: SAS Institute Inc., Available at <http://support.sas.com/techsup/technote/ts407.html>

ACKNOWLEDGMENTS

The author would like to thank Paul Slagle, Carol Matthews, and Andrew Newcomer for reviewing the paper.

CONTACT INFORMATION

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

Lisa Fine

United Biosource Corporation
2200 Commonwealth Blvd., Suite 100
Ann Arbor, MI 48105
Work Phone: (734) 994-8940 x1616
Fax: (734) 994-8927
E-mail: lisa.fine@unitedbiosource.com
Web: www.unitedbiosource.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.