Ordering PROC FREQ Around

Jonathan Kerman, Johns Hopkins University, Baltimore, MD

ABSTRACT

PROC FREQ is one of the most widely used SAS® procedures. The default output of PROC FREQ can compactly summarize large data distributions in one-dimensional, two-dimensional or multi-dimensional frequency tables. Yet the lesser-known ORDER= option gives the user added flexibility in controlling the order of the output. This paper will demonstrate use of the ORDER= option and its four sub-options, showing how you can order PROC FREQ to print the way you want.

INTRODUCTION

PROC FREQ is a trustworthy workhorse in the SAS toolbox. It may be old, but it still runs, simply and reliably. Any user of PROC FREQ will be familiar with output looking like this:

Year	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1998	1317	10.64	1317	10.64
1999	1339	10.82	2656	21.45
2000	1403	11.33	4059	32.79
2001	1249	10.09	5308	42.88
2002	1320	10.66	6628	53.54
2003	1094	8.84	7722	62.37
2004	1328	10.73	9050	73.10
2005	968	7.82	10018	80.92
2006	1266	10.23	11284	91.15
2007	1096	8.85	12380	100.00

Figure 1. **Number of food-borne illness outbreaks, by year**

This example and the ones that follow use data adapted from the Centers for Disease Control (CDC) website of food-borne illnesses (*http://wwwn.cdc.gov/foodborneoutbreaks*). This distribution shows the number of food-borne illness outbreaks by year from 1998 to 2007.*

But suppose you want to change the order of your output, for example, by displaying the most recent year first. PROC FREQ provides an option to give you additional control over the order of the output.

Before we discuss changing the sorting order, let's review how SAS orders things by default. The alphanumeric sorting sequence, based on the ASCII system, sorts characters in the following order:

0 – 9, A – Z, a – z,

that is, numerals first, then capital letters, followed by lowercase letters. We'll see how this sorting order affects output in some of the examples below. And we'll return to the exercise of printing the number of outbreaks from latest year to earliest year at the end of the paper.

THE "ORDER=" OPTION

PROC FREQ has an option-- "ORDER=". This option takes four possible values: DATA, FORMATTED, FREQ and INTERNAL. Let's examine what each of these values does.

---

* The example dataset, named *foodill*, includes these fields: *Year* – year of outbreak*; Month* – month of outbreak; *Location*– site where outbreak occurred (e.g.- home, school); *Total_Ill* – number falling ill

ORDER=INTERNAL

First, you should know that even without specifying it, PROC FREQ uses option ORDER=INTERNAL by default. This means that the results are printed in the sorting order of the raw values—the *internal* representation of the variables.

The output in Figure 1 above shows this. The years are displayed in numerical order, just as you would expect. The program that generated this output could be written relying on the default behavior:

```
proc freq data=foodill;
table year;
run;
```

or by stating the option explicitly:

```
proc freq data=foodill order=internal;
table year;
run;
```

In the discussion below, I'll show the default output of several more variables. In these examples, I'll sometimes specify the ORDER=INTERNAL option and sometimes leave it out to emphasize that either way produces output sorted by the internal values.

ORDER=FREQ

The option ORDER=FREQ sorts the output by frequency of occurrence, from most common to least common. This can be particularly useful if you have many values and you want to easily see which ones occur with the greatest frequency. As an example, let's look at the distribution of settings where food-borne illnesses broke out.

Using the default ordering, here is the program to display the number of outbreaks by the place where they occurred:

```
proc freq data=foodill;
table location;
run;
```

The output looks like this:

Location	Frequency	Percent	Cumulative Frequency	Cumulative Percent
banquet	185	1.55	185	1.55
camp	104	0.87	289	2.43
church	226	1.90	515	4.32
day care	41	0.34	556	4.67
fair	74	0.62	630	5.29
grocery	14	0.12	644	5.40
home	2480	20.81	3124	26.22
hospital	74	0.62	3198	26.84
office	120	1.01	3318	27.85
other	1341	11.25	4659	39.10
picnic	125	1.05	4784	40.15
prison	127	1.07	4911	41.22
restaurant	5861	49.19	10772	90.41
school	477	4.00	11249	94.41
unknown	284	2.38	11533	96.79
wedding	33	0.28	11566	97.07
work (not caf)	331	2.78	11897	99.85
work cafeteria	18	0.15	11915	100.00

Frequency Missing = 465

Figure 2a. **Number of food-borne illness outbreaks, by location**

This output is fine, but it takes some work to look through and find the locations where the most or the fewest outbreaks occurred. Instead, we can run the program with option ORDER=FREQ:

```
proc freq data=foodill order=freq;
table location;
run;
```

Now, the output will look like this:

Location	Frequency	Percent	Cumulative Frequency	Cumulative Percent
restaurant	5861	49.19	5861	49.19
home	2480	20.81	8341	70.00
other	1341	11.25	9682	81.26
school	477	4.00	10159	85.26
work (not caf)	331	2.78	10490	88.04
unknown	284	2.38	10774	90.42
church	226	1.90	11000	92.32
banquet	185	1.55	11185	93.87
prison	127	1.07	11312	94.94
picnic	125	1.05	11437	95.99
office	120	1.01	11557	97.00
camp	104	0.87	11661	97.87
fair	74	0.62	11735	98.49
hospital	74	0.62	11809	99.11
day care	41	0.34	11850	99.45
wedding	33	0.28	11883	99.73
work cafeteria	18	0.15	11901	99.88
grocery	14	0.12	11915	100.00

Frequency Missing = 465

Figure 2b. **Number of food-borne illness outbreaks, by location, ORDER=FREQ**

This display shows us clearly where most outbreaks occur. Nearly half of all food-related incidents happened in restaurants. We can also see that together, home and restaurant settings account for 70% of all reported outbreaks. These statistics and similar groupings are readily apparent from the re-ordered output.

ORDER=FORMATTED

While the default option prints output in order of the internal values, ORDER=FORMATTED displays output in the sorting order of the *formats* applied to the variables. Obviously, in order to demonstrate this option, we will need to use a variable with SAS formats applied.*

Variable Total_Ill records the number of people falling ill at each outbreak. First, let's display the default output without using any formats. Here is the program:

```
proc freq data=foodill order=internal;
table total_ill;
run;
```

* Using option ORDER=FORMATTED with an unformatted variable has no effect.

And a partial output would look like this:

Total_Ill	Frequency	Percent	Cumulative Frequency	Cumulative Percent
2	2094	16.91	2094	16.91
3	1336	10.79	3430	27.71
4	1054	8.51	4484	36.22
5	670	5.41	5154	41.63
6	603	4.87	5757	46.50
7	462	3.73	6219	50.23
8	408	3.30	6627	53.53
9	349	2.82	6976	56.35
10	322	2.60	7298	58.95
.				
.				
.				
1200	1	0.01	12379	99.99
1644	1	0.01	12380	100.00

Figure 3a. **Number of food-borne illness outbreaks, by number falling ill (*excerpt*)**

The complete output runs over 250 lines. Clearly, this variable would best be displayed by summarizing the raw values, so we'll create a format to break the numbers down into categories.

Suppose the CDC had rated outbreaks by their severity. They might assign the following scheme:

Category 1 – 1000 or more ill
Category 2 – 100 to 999 ill
Category 3 – 25 to 99 ill
Category 4 – 1 to 24 ill

And we can create a format for these categories:

```
proc format;
value ctg 1000-high='Category 1 (>= 1000)'
          100-999  ='Category 2 (100-999)'
          25-99    ='Category 3 (25-99)'
          1-24     ='Category 4 (1-24)';
run;
```

If we run PROC FREQ using our new format, the default output still uses the internal values to order the output. The formats representing the smaller numbers print first, and the rows are ordered from 'Category 4' down to 'Category 1'. Here is the program and the output:

```
proc freq data=foodill;
table total_ill;
format total_ill ctg.;
run;
```

Total_Ill	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Category 4 (1-24)	9877	79.78	9877	79.78
Category 3 (25-99)	2124	17.16	12001	96.94
Category 2 (100-999)	377	3.05	12378	99.98
Category 1 (>= 1000)	2	0.02	12380	100.00

Figure 3b. **Number of food-borne illness outbreaks, by number falling ill, formatted**

If we use the option ORDER=FORMATTED, we have the following program and output:

```
proc freq data=foodill order=formatted;
table total_ill;
format total_ill ctg.;
run;
```

Total_Ill	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Category 1 (>= 1000)	2	0.02	2	0.02
Category 2 (100-999)	377	3.05	379	3.06
Category 3 (25-99)	2124	17.16	2503	20.22
Category 4 (1-24)	9877	79.78	12380	100.00

Figure 3c. **Number of food-borne illness outbreaks, by number falling ill, formatted, ORDER=FORMATTED**

The rows are now printed in the alphanumeric order of the format's *label*. As a result, the categories are listed in a more logical manner.

ORDER=DATA

The final sub-option prints output in the order that the values are *encountered in the dataset*. At first glance, this might not seem so helpful.

Let's print the first ten lines of dataset *foodill*, noticing the arbitrary order of values for variable Total_Ill:

Obs	Year	Month	Location	Total_ Ill
1	1998	January		218
2	1998	January	home	2
3	1998	January		8
4	1998	January	home	3
5	1998	January	home	2
6	1998	January	other	26
7	1998	January		8
8	1998	January	home	4
9	1998	January		2
10	1998	January	day care	6

Figure 4. **First ten lines of dataset *foodill*, selected variables displayed**

Now, suppose we run PROC FREQ on variable Total_Ill, using option ORDER=DATA. Here is the program and the first few lines of output:

```
proc freq data=foodill order=data;
table total_ill;
run;
```

Total_Ill	Frequency	Percent	Cumulative Frequency	Cumulative Percent
218	1	0.01	1	0.01
2	2094	16.91	2095	16.92
8	408	3.30	2503	20.22
3	1336	10.79	3839	31.01
26	101	0.82	3940	31.83
4	1054	8.51	4994	40.34
6	603	4.87	5597	45.21

Figure 5. **Number of food-borne illness outbreaks, by number falling ill, ORDER=DATA (*excerpt*)**

The values are indeed tabulated and printed in the order they appear in the dataset. But this output doesn't seem helpful at all. Why, you might wonder, would I ever want to use ORDER=DATA? Believe it or not, there are times when this option can come in handy. Here's an example.

Let's display the months when food-borne outbreaks occurred. The program for this task is:

```
proc freq data=foodill;
table month;
run;
```

And the default output would look like this:

Month	Frequency	Percent	Cumulative Frequency	Cumulative Percent
April	1115	9.01	1115	9.01
August	996	8.05	2111	17.05
December	1184	9.56	3295	26.62
February	929	7.50	4224	34.12
January	944	7.63	5168	41.74
July	1016	8.21	6184	49.95
June	1172	9.47	7356	59.42
March	1052	8.50	8408	67.92
May	1231	9.94	9639	77.86
November	1030	8.32	10669	86.18
October	905	7.31	11574	93.49
September	806	6.51	12380	100.00

Figure 6a. **Number of food-borne illness outbreaks, by month**

The months, which are stored as character strings in dataset *foodill*, print in alphabetical order—not the output we were expecting. It would be much more helpful to have month print in calendar order.

Suppose we know that the records in our dataset are in chronological order (beginning with January) and that all months have some outbreaks. (In other words, records for January 1998 come first, followed by those from February 1998, March 1998, etc.) We can use these facts along with option ORDER=DATA to print the output in a more intuitive order.

First, here's the program:

```
proc freq data=foodill order=data;
table month;
run;
```

And the new output is:

Month	Frequency	Percent	Cumulative Frequency	Cumulative Percent
January	944	7.63	944	7.63
February	929	7.50	1873	15.13
March	1052	8.50	2925	23.63
April	1115	9.01	4040	32.63
May	1231	9.94	5271	42.58
June	1172	9.47	6443	52.04
July	1016	8.21	7459	60.25
August	996	8.05	8455	68.30
September	806	6.51	9261	74.81
October	905	7.31	10166	82.12
November	1030	8.32	11196	90.44
December	1184	9.56	12380	100.00

Figure 6b. **Number of food-borne illness outbreaks, by month, ORDER=DATA**

FORMATTED VARIABLES VS. ORDER=FORMATTED

It's important to keep in mind the distinction between using formatted variables and using the option ORDER=FORMATTED. Displaying formatted variables while using the default behavior of PROC FREQ keeps the internal ordering; it just prints using the label given in the format.

To illustrate this, let's create a new format for the number of people becoming ill at an outbreak.

```
proc format;
value n_ill 1-9      ='Fewer than 10'
            10-49    ='10-49'
            50-99    ='50-99'
            100-250  ='100-250'
            251-high ='Greater than 250';
run;
```

Here, we run PROC FREQ formatting variable Total_Ill and relying on the default ordering:

```
proc freq data=foodill;
table total_ill;
format total_ill n_ill.;
run;
```

The output produced is:

Total_Ill	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Fewer than 10	6976	56.35	6976	56.35
10-49	4372	35.32	11348	91.66
50-99	653	5.27	12001	96.94
100-250	304	2.46	12305	99.39
Greater than 250	75	0.61	12380	100.00

Figure 7a. **Number of food-borne illness outbreaks, by number falling ill, formatted**

This output is useful and straightforward. But suppose we had *also* specified option ORDER=FORMATTED. Here is the program and the output:

```
proc freq data=foodill order=formatted;
table total_ill;
format total_ill n_ill.;
run;
```

The output produced is:

Total_Ill	Frequency	Percent	Cumulative Frequency	Cumulative Percent
10-49	4372	35.32	4372	35.32
100-250	304	2.46	4676	37.77
50-99	653	5.27	5329	43.05
Fewer than 10	6976	56.35	12305	99.39
Greater than 250	75	0.61	12380	100.00

Figure 7b. **Number of food-borne illness outbreaks, by number falling ill, formatted, ORDER=FORMATTED**

This output is not very user-friendly. PROC FREQ does use the formats—so that we don't have 250-plus lines of output—but the output rows are sorted by the *label* used for the format, in strict ASCII order. This is one example of when *not* to use ORDER=FORMATTED.

As another example, let's create a format to represent the months as calendar quarters. Because the months are character variables, we must create a character format (beginning with '$'):

```
proc format;
value $qtr 'January','February','March'    = 'Q1'
           'April','May','June'            = 'Q2'
           'July','August','September'     = 'Q3'
           'October','November','December' = 'Q4';
run;
```

First, let's run PROC FREQ using the format together with the default ordering:

```
proc freq data=foodill order=internal;
table month;
format month $qtr.;
run;
```

Month	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Q2	3518	28.42	3518	28.42
Q3	2818	22.76	6336	51.18
Q4	3119	25.19	9455	76.37
Q1	2925	23.63	12380	100.00

Figure 8a. **Number of food-borne illness outbreaks, by quarter, formatted**

Why did PROC FREQ display the output rows in this order? Because, even though it's the labels that are being printed, the underlying order is still the *internal* value of the variable. This means that behind the "mask" of, 'Q1', 'Q2', 'Q3' and 'Q4', PROC FREQ still wants to order the rows "April' (Q2), then 'August' (Q3), then 'December' (Q4), etc.

In order to get the output sorted by quarter, one way is to use ORDER=FORMATTED:

```
proc freq data=foodill order=formatted;
table month;
format month $qtr.;
run;
```

Month	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Q1	2925	23.63	2925	23.63
Q2	3518	28.42	6443	52.04
Q3	2818	22.76	9261	74.81
Q4	3119	25.19	12380	100.00

Figure 8b. **Number of food-borne illness outbreaks, by quarter, formatted, ORDER=FORMATTED**

A second way, again assuming that the records in our dataset are in chronological order beginning with 'January', is to use ORDER=DATA. This option will produce output identical to Figure 11 (because 'January", represented by 'Q1', appears first in the dataset, and so on.)

In summary, then, formats use a different *text* to display the output—'Q3' instead of 'September'; 'Fewer than 10' instead of '6'—but formats do nothing to change the order of the output. Option ORDER=FORMATTED, on the other hand, will change the order of the output. You will probably only want to use this option when the formats themselves have a meaningful alphanumeric order.

FORCING SAS TO PRINT IN REVERSE ORDER

After seeing all of these options in use, we can go back to the example mentioned at the beginning—displaying the distribution of food-borne illness outbreaks by year, but in reverse order.

Here, we can use a bit of a trick. Knowing that ORDER=DATA will print in the order that each value is encountered in the dataset, we can first force a reverse sort:

```
proc sort data=foodill;
by descending year;
run;
```

Then use ORDER=DATA to get the output we want:

```
proc freq data=foodill order=data;
table year;
run;
```

And the re-ordered output is:

Year	Frequency	Percent	Cumulative Frequency	Cumulative Percent
2007	1096	8.85	1096	8.85
2006	1266	10.23	2362	19.08
2005	968	7.82	3330	26.90
2004	1328	10.73	4658	37.63
2003	1094	8.84	5752	46.46
2002	1320	10.66	7072	57.12
2001	1249	10.09	8321	67.21
2000	1403	11.33	9724	78.55
1999	1339	10.82	11063	89.36
1998	1317	10.64	12380	100.00

Figure 9. **Number of food-borne illness outbreaks, by year, most recent year first**

CONCLUSIONS

With the "ORDER=" option, PROC FREQ provides a method to modify the order of its output. The values that this option can take-- DATA, FORMATTED, FREQ and, the default, INTERNAL—along with a clear understanding of how each one functions provide the user with tools to help create clearer frequency tables.

ACKNOWLEDGMENTS

CONTACT INFORMATION

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

Author Name: Jonathan Kerman
Company: Johns Hopkins University, School of Public Health
Address: 615 North Wolfe Street, Room E7008
City, State ZIP: Baltimore, Maryland, 21205
Work Phone: (410) 955-4583
Fax: (410) 955-7587
Email: jkerman@jhmi.edu