



THE
POWER
TO KNOW.

Technical Paper

What Happened to My Data?

Understanding how results are affected when formatting is applied to classification variables

Table of contents

Introduction	1
Grouping observations with one classification variable	1
Expected results	3
Grouping observations with multiple classification variables	5
Unexpected results	7
Conclusion.....	8
Resources	8

Introduction

Classification variables are used to group observations that have the same values. In the REPORT procedure, ORDER, GROUP, and ACROSS variables are classification variables. In the TABULATE, SUMMARY, and MEANS procedures, a CLASS variable is a classification variable. Because the classification process groups individual values, when a formatted value is used for classification the results might be unexpected.

This paper explains how classification variable values are handled when a FORMAT statement or FORMAT option is applied. Topics include grouping observations with one classification variable and with multiple classification variables. The supporting examples focus on Base SAS® reporting procedures, PROC REPORT, PROC TABULATE, PROC SUMMARY, and PROC MEANS.

Grouping observations with one classification variable

The following simple PROC REPORT example shows how the order of the output changes when a format is applied--the grouping occurs around the formatted value. Notice how the order of the values for 'f' and 'g' changes. The output is shown in Display 1.

```
proc format;
value $myfmt 'g' = 'one'
             'f' = 'two';
run;

data test;
input myval $ mynum;
cards;
f 1
f 3
f 5
g 2
g 4
g 6
;
run;

proc report data=test nowd;
column myval mynum;
  define myval / group format=$myfmt3.;
/* order=formatted is the default ordering of proc report */
run;
```

Here is a comparable PROC TABULATE example. Note that the ORDER=FORMATTED option must be added because the default order is INTERNAL for all reporting procedures other than PROC REPORT.

```
proc tabulate data=test order=formatted;
class myval;
var mynum;
format myval $myfmt3.;
table myval, mynum;
run;
```

Because the variable MYVAL is a classification variable, all equivalent values are grouped together. In this case, a format is applied so the result is two distinct formatted values for MYVAL, "one" and "two". The output is shown in Display 1.

myval	mynum
one	12
two	9

Display 1. PROC REPORT and PROC TABULATE Output Using a FORMAT Statement

Internally, a separate mapping table is formed for each class variable. For each unique formatted value that is encountered during the input data processing, a new row is added to the mapping table. This includes the row number, the formatted value, and the internal value that is associated with the formatted value and is in the order of the data. The output in Display 2 is an example of the class mapping table for *myval*.

R_num	Fmt_val	Int_val
1	'two'	'f'
2	'one'	'g'

Display 2. Output of the Class Mapping Table *myval*

Notice that "two" is the first formatted entry because 'f' occurs before 'g' in the input data set.

In addition to associating the formatted value with each mapping table row, the procedure also keeps or retains the unformatted (internal) value.

If the ORDER=INTERNAL option is used, as shown in the following PROC REPORT example, then the ordering of the variable MYVAL is based on the unformatted value.

```
proc report data=test nowd;
column myval mynum;
define myval / group format=$myfmt3. order=internal;
run;
```

Here is a comparable PROC TABULATE code.

```
proc tabulate data=test;
class myval;
var mynum;
format myval $myfmt3.;
table myval, mynum;
run;
```

In this case, because the internal value of 'f' comes before 'g', the output would look like Display 3:

myval	mynum
two	9
one	12

Display 3. PROC REPORT Output Using the ORDER=INTERNAL Option

Now let's make the example a little more complicated by having two different data values that use the same formatted value.

```
proc format;
value $myfmt 'd' = 'one'
              'e' = 'two'
              'f' = 'two'
              'g' = 'one';
run;

data test;
input myval $ mynum;
cards;
f 1
f 3
f 5
g 2
g 4
g 6
e 2
e 4
e 6
d 7
d 9
d 11
;
run;

proc report data=test nowd;
column myval mynum;
define myval / group format=$myfmt3.;
run;
```

Again, there are only two distinct formatted values, 'one' and 'two'. The observations are consolidated into those two distinct mapping table rows as shown in the output in Display 4.

myval	mynum
one	39
two	21

Display 4. PROC REPORT Output for Two Distinct Formatted Values

Expected results

In the first example, we stated that the unformatted value is stored with each mapping table row. What value is stored when there are two or more unformatted values for a mapping table row? The procedure cannot keep all of the unformatted values; instead, it keeps only one unformatted value for each row in the mapping table. The rule, therefore, is that the *lowest* unformatted value is kept for that mapping table row, where *lowest* is determined by the ORDER= option.

For the preceding example, 'd' is kept for the 'one' mapping table row, and 'e' is kept for the 'two' mapping table row.

In PROC REPORT, if the ORDER=INTERNAL option is requested, the 'one' mapping table row would again print first because the unformatted value 'd' comes before 'e'. The output is shown in Display 5.

```
proc report data=test nowd;
column myval mynum;
  define myval / group format=$myfmt3. order=internal;
run;
```

myval	mynum
one	39
two	21

Display 5. PROC REPORT Output Using the ORDER=INTERNAL Option

Let's change the data to verify that.

```
proc format;
value $myfmt 'd' = 'one'
              'c' = 'two'
              'f' = 'two'
              'g' = 'one' ;
run;

data test;
input myval $ mynum;
cards;
f 1
f 3
f 5
g 2
g 4
g 6
c 2
c 4
c 6
d 7
d 9
d 11
;
run;

proc report data=test nowd;
column myval mynum;
  define myval / group format=$myfmt3.;
run;

proc report data=test nowd;
column myval mynum;
  define myval / group format=$myfmt3. order=internal;
run;
```

Here is a comparable PROC TABULATE example.

```
proc tabulate data=test order=formatted;
class myval;
var mynum;
format myval $myfmt3.;
table myval, mynum;
run;

proc tabulate data=test;
class myval;
var mynum;
format myval $myfmt3.;
table myval, mynum;
run;
```

The output using the ORDER=FORMATTED option is shown in Display 6. The output using the ORDER=INTERNAL option is shown in Display 7.

myval	mynum
one	39
two	21

Display 6. PROC REPORT and PROC TABULATE Output Using the ORDER=FORMATTED Option

myval	mynum
two	21
one	39

Display 7. PROC REPORT and PROC TABULATE Output Using the ORDER=INTERNAL Option

There are still only two mapping table rows—'one' and 'two'. However, the lowest value for 'one' is 'd' now, and the lowest value for 'two' is 'c'. If the formatted value (the default for PROC REPORT) determines the order of your output, then 'one' prints first. If ORDER=INTERNAL is specified, then 'two' would print first.

Now let's consider another example.

Grouping observations with multiple classification variables

In this example, there are two classification variables, YR and MON_YR. There are two distinct levels for YR, 2005 and 2006, and five distinct levels for MON_YR. Formatted as MONNAME3, the values are Aug, Sep, Oct, Nov, and Dec.

```
data test;
input yr mon_yr;
cards;
2005 16771
2006 17014
2006 17045
2006 17075
2006 17106
2006 17136
;
run;
```

```

proc report nowd data=test out=myoutput;
column yr mon_yr;
  define yr / group;
  define mon_yr / group format=monname3. order=internal;
run;

```

Here is a comparable PROC TABULATE example.

```

proc tabulate data=test;
class yr mon_yr;
format mon_yr monname3.;
table yr*mon_yr, n;
run;

```

The output is shown in Display 8.

yr	mon_yr
2005	Dec
2006	Dec
	Aug
	Sep
	Oct
	Nov

Display 8. PROC REPORT and PROC TABULATE Output Using the ORDER=INTERNAL Option

There are two internal values for the mapping table row Dec: 16771 and 17136. Because the lowest unformatted value is kept when there are two or more unformatted values in the same mapping table row, the value 16771 is kept for 'Dec'.

The associated internal values for MON_YR are:

```

"Aug"      17014
"Sep"      17045
"Oct"      17075
"Nov"      17106
"Dec"      16771

```

The associated internal values for YR are:

```

2005      2005
2006      2006

```

Now that the mapping table rows have been created, the procedure orders the columns and rows. Because ORDER=INTERNAL was specified, values are ordered using the internal values that are associated with each mapping table row.

Unexpected results

When YR=2005, there is only one MON_YR value, so no ordering needs to take place. However, when YR=2006, the MON_YR column has the following ordered values: 16771, 17014, 17045, 17075, and 17106. When the report is printed, the formatted value is printed in this order with the formatted value shown. As a result, the months do not appear in chronological order.

When you specify an output data set with PROC REPORT, the unformatted value is shown. Display 9 shows a PROC PRINT of the output data set MYOUTPUT from PROC REPORT:

yr	mon_yr	_BREAK_
2005	16771	
2006	16771	
2006	17014	
2006	17045	
2006	17075	
2006	17106	

Display 9. PROC REPORT Output Showing an Unformatted Value

Unexpected output can also occur when the classification variable is numeric and it is grouped using a format with less precision than the internal values.

In the following code, look at the value for CFB in the input data set. There are three different values.

```
data temp_ds;
input cfb;
cards;
.
0.6594
.
0.6629
;
run;

proc report data=temp_ds nowd missing out=test2;
column cfb;
define cfb / order format=6.3;
run;

proc print;
run;
```

Because CFB is a classification variable and has a format, there is a mapping table row for the formatted value of 0.066 and the missing value. The lowest unformatted value is 0.06594 and is kept with the formatted value 0.066 mapping table row. This output is shown in Display 10.

cfb
.
0.066

Display 10. PROC PRINT of PROC REPORT Output Showing an Unformatted Value

The output data set, TEST2, contains the lowest value of CFB. Display 11 shows a PROC PRINT output for TEST2.

```
   cfb      _BREAK_
   .
   .
   0.06594
   0.06594
```

Display 11. PROC PRINT of PROC REPORT Output

By default, PROC TABULATE and PROC SUMMARY display the formatted value in the output data set. The internal values are the same as those displayed in PROC REPORT.

Conclusion

In summary, classification variables are used to group data values that are the same. When a FORMAT statement or FORMAT option is applied and causes more than one value of a variable to be in a mapping table row, then only the lowest unformatted value is kept with the associated bucket. Understanding this behavior will clarify unexpected results for SAS® reporting procedures.

Resources

Basic information on the REPORT, TABULATE, SUMMARY, and MEANS procedures is available in the following resource:

SAS Institute Inc. 2006. *SAS OnlineDoc® 9.1.3*. Cary, NC: SAS Institute Inc. Available at support.sas.com/onlinedoc/913/docMainpage.jsp.

Questions

If you have questions about the content in this paper send e-mail to support@sas.com.

