

Multiple Facts about Multilabel Formats

Gwen D. Babcock, New York State Department of Health, Troy, NY

ABSTRACT

PROC FORMAT is a powerful procedure which allows the viewing and summarizing of data in various ways without creating new variables in the data step. The addition of the MULTILABEL option in SAS® 8 has made PROC FORMAT even more powerful by allowing formats to have duplicate and/or overlapping ranges. This paper will give examples of the use of MULTILABEL formats with the /mlf option in PROC MEANS, PROC SUMMARY, and PROC TABULATE. In addition, this paper addresses what happens when MULTILABEL formats are used in other procedures or without the /mlf option. This paper also shows how MULTILABEL formats, like other formats, can be created from an existing dataset. MULTILABEL formats are a valuable addition to any SAS® programmer's repertoire.

INTRODUCTION

Many times, the way the data are stored in a dataset is not the way you wish to display it. In this case, PROC FORMAT can be used to group and recode variables. Using proc format, you can label variable values. The labels are then displayed in procedure output instead of the underlying values. Traditional formats allow only one label per data value. In contrast, multilabel formats allow more than one label per data value. This allows for the creation of overlapping groups (such as children and teenagers) and total summaries for data analysis. The drawback of multilabel formats is that they can only be used with a limited number of procedures. Three procedures that it can be used with are PROC MEANS, PROC SUMMARY, and PROC TABULATE.

This paper shows how to use multilabel formats, with special emphasis on frequency analysis, and what happens when you forget and use a multilabel format for ordinary analysis. The output is from SAS® 9.1.2 running in Windows XP.

HOW TO USE FORMATS AND MULTILABEL FORMATS

This is an example of a format. The original data is either '1' or '2', which represents 'Male' and 'Female', respectively.

```
data nesug;
input sex $1.;
datalines;
1
2
1

1
1
2
;
run;

proc format;
value $sex
'1'='Male'
'2'='Female';
quit;

proc freq data=nesug;
table sex;
format sex $sex.;run;
```

Here is the output. The '1's and '2's have been changed to 'Male' and 'Female'.

The FREQ Procedure

sex	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Male	4	66.67	4	66.67
Female	2	33.33	6	100.00

Frequency Missing =1

What if we wanted to display both the number of males and females and the total number of people? We could use a multilabel format, like this:

```
proc format;
value $gender (multilabel)
'1'='Male'
'2'='Female'
'1','2',' '='Total people';
quit;
```

But this format gives the exact same results as the previous one when we run the PROC FREQ. Why? Because PROC FREQ doesn't understand multilabel formats. Only PROC MEANS, PROC SUMMARY, and PROC TABULATE can use multilabel formats. We can use PROC MEANS to determine the frequencies we want:

```
proc means data=nesug missing;
class sex /mlf;
*when all variables are character variables, proc means produces a simple count
of observations;
format sex $gender.;
run;
```

Note that in addition to using our new multilabel format, we must also use the MLF option in the CLASS statement. This option tells the procedure that the format for the class variable will be multilabel. Here is the output:

The MEANS Procedure

sex	N Obs
Female	2
Male	4
Total people	7

PROC SUMMARY is very similar to PROC MEANS. However, PROC MEANS will print output by default, while PROC SUMMARY will not. If you omit the VAR statement, PROC MEANS analyzes all numeric variables not included in other statements, while PROC SUMMARY produces a count.

Multilabel formats can also be used in PROC TABULATE. Here is some example code and its output:

```
proc tabulate data=nesug missing;
  class sex /mlf; /*the CLASS statement identifies character variables*/
  /*a var statement would identify numeric variables, if we had any*/
  table sex,n;
  format sex $gender.;
run;
```

	N
sex	
Female	2.00
Male	4.00
Total people	7.00

Like PROC MEANS, PROC TABULATE uses the MLF option in the CLASS statement to tell the procedure that the format for the class variable will be multilabel.

TO ERR IS HUMAN

Multilabel formats can be used with PROC MEANS, PROC SUMMARY, and PROC TABULATE. All three of these procedures require you to use the MLF option to use multilabel formats. But what happens when we forget these three little letters? How does it affect our results?

TO FORGIVE IS SAS®

Consider the following code, where we define a multilabel format and then fail to use the MLF option in PROC MEANS. The output follows:

```
proc format;
  value $gender (multilabel)
    '1'='Male'
    '2'='Female'
    '1','2',' '='Total people';
quit;

proc means data=nesug;
  class sex;
  format sex $gender.;
run;
```

The MEANS Procedure

	N
sex	Obs
Male	4
Female	2

It appears that SAS® is forgiving; even though our totals are not displayed, we still get the correct values for males and females. I will now continue to explore to see if this is a general rule.

Consider the following code, which is similar to the previous section, except that we change the order in the PROC FORMAT value statement.

```
proc format;
value $gender (multilabel)
'1','2',' '='Both sexes'
'1'='Male'
'2'='Female';
quit;

proc means data=nesug missing;
class sex;
format sex $gender.;
run;
```

The MEANS Procedure

	N
sex	Obs
Both sexes	7

Changing the order in the VALUE statement of PROC FORMAT gives completely different results. SAS® now reports the total population rather than counting by sex. SAS® appears to assign the first label it encounters in format to the data value, and ignores all subsequent labels.

TO NOT FORGIVE IS ALSO SAS®

Consider this code, in which we again change the order in the value statement of the PROC FORMAT procedure:

```
proc format;
value $gender (multilabel)
'1'='Male'
'1','2',' '='Both sexes'
'2'='Female';
quit;

proc means data=nesug missing;
class sex;
format sex $gender.;
run;
```

The MEANS Procedure

	N
sex	Obs
Both sexes	3
Male	4

In this case, we get incorrect results because we used a multilabel format without using the MLF option in the CLASS statement. SAS® appears to first assign the label "Male" to the value '1', then it assigns the value 'Both Sexes' to '2' and ' '; it can't use this again for '1', because '1' already has a label. Similarly, once '2' is assigned the label 'Both Sexes', it cannot subsequently be assigned the label 'Female'. Therefore, when writing multilabel formats, one should consider whether these formats may be accidentally or intentionally used without the MLF option or in procedures that do not support multilabel formats. If so, consideration should be given to the order of the assignments in the VALUE statement so that the desired results (or at least not incorrect results) are produced.

NUMERIC MULTILABEL FORMATS USED WITHOUT /MLF

SAS® handles numeric multilabel formats differently than character multilabel formats. Consider the following code, which is designed to break up observations into overlapping age groups and provide counts in each age group:

```

data nesug2;
input age;
datalines;
5
10
3
54
80
;
run;quit;

proc format;
value age (multilabel)
1-4='Preschool'
1-18='Children'
19-120='Adults';
run;quit;

proc means data=nesug2 n;
title1 "Results using the /MLF option";
class age /mlf;
var age;
format age age.;
run;

proc means data=nesug2 n;
title1 "Results without the /MLF option";
class age;
var age;
format age age.;run;
title1;

```

Results using the /MLF option

The MEANS Procedure

Analysis Variable : age

age	N	
	Obs	N
Adults	2	2
Children	3	3
Preschool	1	1

Results without the /MLF option

The MEANS Procedure

Analysis Variable : age

age	N	
	Obs	N
Preschool	1	1
Children	2	2
Adults	2	2

Without the /MLF option, the procedure gives incorrect results, since the preschoolers are not included in the 'Children' category. Can we fix this by changing the order of our statements in PROC FORMAT? Consider the following code, where we put the 'Adults' category first, followed by 'Children' and then 'Preschool':

```

/*does order matter?*/
proc format;
value age (multilabel)
19-120='Adults'
1-18='Children'
1-4='Preschool';
quit;

proc means data=nesug2 n;
class age;
var age;
format age age.;
run;

```

Here are the results:

The MEANS Procedure

Analysis Variable : age

age	N	
	Obs	N
Preschool	1	1
Children	2	2
Adults	2	2

It appears that SAS® sorts the format labels by value and then assigns them. So it doesn't matter what order you type the labels in, unless...you use the NOTSORTED options in PROC FORMAT. Consider this example:

```

/*it's the internal order that's important*/
proc format;
value age (multilabel notsorted)
19-120='Adults'
1-18='Children'
1-4='Preschool';
quit;

proc means data=nesug2 n;
class age;
var age;
format age age.;run;

```

The MEANS Procedure

Analysis Variable : age

age	N	
	Obs	N
Children	3	3
Adults	2	2

Although these results lack the category 'Preschoolers', the results that do display are correct, even though we did not use the /MLF option in PROC MEANS. The NOTSORTED option in PROC FORMAT forces SAS® to assign the values in the order we typed them, not in numerical sort order.

MULTILABEL FORMATS WITH CNTLIN DATASETS

Typing in long value statements can be tedious. Fortunately, SAS® provides a way to create formats from datasets using the CNTLIN= option in the PROC FORMAT statement. These datasets are called 'input control' datasets, and each one can contain one or more formats. Consider the following example:

```

data forformat;
fmtname="$gend"; hlo="M"; format label $10.;
start="1"; label="Male"; output;
start="2"; label="Female"; output;
start="1"; label="Both sexes"; output;
start="2"; label="Both sexes"; output;
start=" "; label="Both sexes"; output;
run;

```

The essential parts of the format are present in the dataset. The 'start' variable contains the values of the variable that are to be formatted. The 'label' variable tells how the 'start' value should be displayed by SAS® when the format is used. The 'fmtname' variable contains the name of the format. One input control dataset can contain multiple formats. These variables are the same for both ordinary and multilabel formats. In addition to these variables, the variable 'hlo' is used to tell SAS® that this is a multilabel format. To declare that this is a multilabel format, the 'hlo' variable is set to 'M'.

Once the dataset is complete, it can be used with the 'cntlin' option of PROC FORMAT to create the actual format. Then the format can be used as usual. Here is an example of how the previous dataset could be used:

```

proc format cntlin=forformat;
quit;

proc means data=nesug missing;
class sex /mlf;
format sex $gend.;
run;

```

CONCLUSIONS

Multilabel formats are a powerful way to control how data are organized and displayed for presentation. They can be typed in or created using input control datasets. However, multilabel formats also have the potential to provide incorrect results if used improperly. Therefore, SAS® programmers would do well to arrange the VALUE statements in PROC format so that if they do forget to use the /mlf option or use a multilabel format in a procedure that does not support it, the results will still be correct, though incomplete. This provides an additional precaution against human error.

ACKNOWLEDGMENTS

Thanks to Thomas Talbot and Sanjaya Kumar for their thoughtful review of this paper.

CONTACT INFORMATION

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

Gwen D. Babcock
New York State Department of Health
547 River St
Troy NY 12180-2216
Work Phone: 518-402-7785
Fax: 518-402-7959
Email: gdb02@health.state.ny.us

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 registered trademarks or trademarks of their respective companies.