

Labels, Labels, and More Labels

Stephanie R. Thompson, Rochester Institute of Technology, Rochester, NY

ABSTRACT

SAS® datasets include labels as optional variable attributes in the descriptor portion. Labels are not a nuisance but something that you can get a lot of use out of. Learn how to get the most out of them in your programs. Here a just a few of the topics that will be covered: how labels can make reports more readable, how to permanently or temporarily create or change labels, how to use them in procedures or not use them, and how to extract them in to a dataset or a macro variable. Consider this paper a primer on the variable label.

INTRODUCTION

Labels can contain useful information about your data. Some procedures use labels automatically which enhances output. Other procedures allow you turn on label use as desired. As useful as this is there may be times when you want to use them in different ways in your program. This paper will cover many aspects of the variable label from creation to extracting them to use in your program in other ways.

CREATING LABELS

Labels can be created as a permanent attribute of the SAS dataset when it is created or modified with a DATA step. This means that subsequent procedures can take advantage of the labels. The maximum length for a variable label is 256 characters. You can tell if a variable has a label or not by looking at the output from PROC CONTENTS. A screenshot of the output of the procedure using sashelp.retail is shown below with the label for the variable sales circled in red.

The CONTENTS Procedure

Data Set Name	SASHELP.RETAIL	Observations	58
Member Type	DATA	Variables	5
Engine	V9	Indexes	0
Created	Wednesday, May 12, 2004 10:54:56 PM	Observation Length	40
Last Modified	Wednesday, May 12, 2004 10:54:56 PM	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label	Retail Sales (Quarterly: 1980Q1-1994Q2)		
Data Representation	WINDOWS_32		
Encoding	us-ascii ASCII (ANSI)		

Engine/Host Dependent Information

Data Set Page Size	4096
Number of Data Set Pages	1
First Data Page	1
Max Obs per Page	101
Obs in First Data Page	58
Number of Data Set Repairs	0
File Name	C:\Program Files\SAS\SAS 9.1\core\sashelp\retail.sas7bdat
Release Created	9.0101M3
Host Created	XP_PRO

Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Format	Label
2	DATE	Num	8	YYQ4.	
5	DAY	Num	8		
4	MONTH	Num	8		
1	SALES	Num	8	DOLLAR10.	Retail sales in millions of \$
3	YEAR	Num	8		

If the dataset you are using does not have a label, you can create them easily in a DATA step. All you need is a label statement. It can be used to create as many labels as you need in one statement. The text in quotes in the example will become the labels for the variables on the left of the equals sign.

```
data newdat;
set dat1;
label var1 = 'Label Number 1'
      var2 = 'Label Two';
run;
```

PROC DATASETS can also be used to create and modify labels. The example below modifies the dataset dat1 by changing (or assigning if they do not already exist) the labels for var1 and var2. The two contents statements have been added since the label statement in PROC DATASETS does not send any messages to the log. The output from the contents statements allows you to see the labels before and after changing the labels.

```
proc datasets;

contents data = newdat;

modify newdat;
label var1 = 'New Label Number 1'
      var2 = 'Other Label Two';

contents data = newdat;

run;
quit;
```

Keep in mind that an assignment statement will create labels, in this example for var1 and var2, if they do not exist but it will overwrite any existing labels. Do not use the label statement in a DATA step or PROC DATASETS if you only want to temporarily change the labels.

Creating a label for temporary use is easy to do. This can be done by inserting a label statement within the procedure. In PROC MEANS below, the label for the variable sales is temporarily changed from 'Retail sales in millions of \$' to 'Sales in Millions.'

```
proc means data = sashelp.retail;
label sales = 'Sales in Millions';
run;
```

USE IN PROCEDURES

Procedures such as PROC FREQ, PROC MEANS, and PROC REPORT automatically use variable labels if they exist. This enhances the output and makes it easier to read. In many instances you will see the variable name and label. Output from the previous PROC MEANS shows how the variable name and label are used.

The MEANS Procedure

Variable	Label	Mean
SALES	Sales in Millions	540.93
DATE		9907.03
YEAR		1986.76
MONTH		5.40
DAY		1.00

However, PROC PRINT is one notable exception. By default, PROC PRINT does not use variable labels. The label option must be used. Only the label will be shown in the output when this option is invoked. Code using the option is shown below:

```
proc print data = sashelp.retail label;
run;
```

Here is the partial output first without the label option then with it.

Obs	SALES	DATE	YEAR	MONTH	DAY
1	\$220	80Q1	1980	1	1
2	\$257	80Q2	1980	4	1
3	\$258	80Q3	1980	7	1

Obs	Retail sales in millions of \$	DATE	YEAR	MONTH	DAY
1	\$220	80Q1	1980	1	1
2	\$257	80Q2	1980	4	1
3	\$258	80Q3	1980	7	1
4	\$295	80Q4	1980	10	1

If you do not like how the label is split when running PROC PRINT, you can switch from the label option to the split= option. The split= option tells SAS to split the label at the character indicated after the equals sign. A space is commonly use but you can also use a letter or special character as shown below.

```
proc print data = sashelp.retail split='*';
label sales = 'Retail sales in*Millions of $';
run;
```

Obs	Retail sales in Millions of \$	DATE	YEAR	MONTH	DAY
1	\$220	80Q1	1980	1	1
2	\$257	80Q2	1980	4	1
3	\$258	80Q3	1980	7	1
4	\$295	80Q4	1980	10	1

Note that the special character does not show up in the output. Use caution when selecting what to split the label with. You could get some interesting results if you are not careful.

EXTRACTING A LABEL INTO A VARIABLE

So far creating and using labels has been covered. Sometimes you may want to do something else with a label. This is not uncommon since labels can contain useful information. Extracting the contents and using them in a different way in a program is one example of another use.

CALL LABEL creates a variable which contains the label of the named variable. It has a very simple syntax that is similar to CALL SYMPUT. This call routine can be inserted into a DATA step and is comparable to an assignment statement.

```
call label(svar, namelabel);
```

Here svar is the name of the variable that has a label that which is being stored in the variable namelabel. It is advisable to set the length of namelabel to accommodate the length of the text in the label. Otherwise the contents of namelabel may be truncated. Since labels can be up to 256 characters, assigning a length longer than that will not serve any purpose.

In the event that svar does not have a label, the name of the variable (svar in this case) will be stored in namelabel. This is a good feature since null values will not be created.

SOME LABEL FUNCTIONS

The VLABEL function performs almost the same as CALL LABEL. One notable difference is that an expression cannot be used as an argument within the function. A direct variable name reference or an array reference can be used as the argument. The form of the function is below.

```
vlabel (varname)
```

Varname is the name of the variable that contains the label you want to assign to another variable. In code, it could be used in an assignment statement such as this.

```
namelabel = vlabel(svar);
```

VLABEL will assign a default length of 200 to any variable that has not previously been assigned a length. If you do not need something that long, assigning variable a length is recommended. On the other hand you may need to increase the length since labels can be up to 256 characters long.

VLABELX is similar to VLABEL in syntax, but it can evaluate an expression to determine the name of the variable whose label will be returned. VLABELX cannot have an array reference as an argument.

PROC CONTENTS OUTPUT

The call routine and labels examples extract labels from one variable at a time. What if you want to extract all labels from all variables? Creating a dataset from PROC CONTENTS will provide a dataset that contains labels as well as other meta data from the dataset. The variable containing the label is named LABEL and it corresponds to the variable listed in NAME. The syntax, suppressing printed output with the noprint option, is below:

```
proc contents noprint data= sashelp.retail out= column_metadata ;  
run;
```

Looking at the PROC PRINT output of column_metadata specifying that only name and label be printed, you can see that null values are created if a variable does not have a label. This is an important difference from CALL LABEL and the VLABEL function.

Obs	NAME	LABEL
1	DATE	
2	DAY	
3	MONTH	
4	SALES	Retail sales in millions of \$
5	YEAR	

CONCLUSION

Labels contain useful information about your data and can be used in a variety of ways. They are simple to create or change. Sometimes their use is automatic and enhances reports. Other times you want to extract that information in other ways. Any way you look at it, there are plenty of options with labels.

ACKNOWLEDGEMENTS

Special thanks to Peter Crawford for his suggestions regarding labels.

CONTACT INFORMATION

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

Stephanie R. Thompson
Rochester Institute of Technology
Nine Lomb Memorial Drive
Rochester, NY 14623-5603
Work Phone: (585) 475-7237
Fax: (585) 475-7950
Email: Stephanie.Thompson@rit.edu
Web: <http://finweb.rit.edu/irps/>

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.