PharmaSUG China 2016 - Paper 81

Macro Quoting: Which Function Should We Use?

Pengfei Guo, MSD R&D (China) Co., Ltd., Shanghai, China

ABSTRACT

There are several macro quoting functions in SAS and even some experienced programmers can't distinguish them exactly. We usually look up the SAS help document when encountering some log issue during using. This article will summarize the difference and list some fallible situations. At the end, we also provide a step-wise method to help user make easy decision.

INTRODUCTION

The macro language is character based. Every word has meaning to the macro processor. But sometimes, the processor needs to deal with some special characters which could make the processor confused. These special characters are not only simple texts but also functions in SAS® program. The purpose of the macro quoting functions is resolving these ambiguities by masking these special characters. We have 7 functions as blew.

%STR/%NRSTR/%QUOTE/%NRQUOTE/%BQUOTE/%NRBQUOTE/%SUPERQ

HOW TO CHOOSE AN APPROPRIATE MACRO?

In summaries, the special characters can divided into the following types.

- Type I:

Operators	+ - * / < > = ¬ ^ | ~
Mnemonics	AND OR NOT EQ NE LE LT GE GT IN
Miscellaneous	blank , ; “” ‘’ () #

- Type II:

Unmatched Characters	“ ‘ ( )

- Type III:

Macro Triggers	& %

Blew Table 1 is a summary of these functions. “ √ ” means this function can deal with this type of characters.

Table 1 Macro quoting functions summary

	Type I	Type II	Type III
%STR	√	√marked with a % sign	
%NRSTR	√	√marked with a % sign	√
%QUOTE	√	√marked with a % sign	
%NRQUOTE	√	√marked with a % sign	√
%BQUOTE	√	√	
%NRBQUOTE	√	√	√
%SUPERQ	√	√	√

Does it seem too complex? You could choose the appropriate function by check the 2 following questions:

- WHEN TO USE A MACRO QUOTING FUNCTION
- DO YOU NEED AMPERSANDS AND PERCENT SIGNS BE RESOLVED?

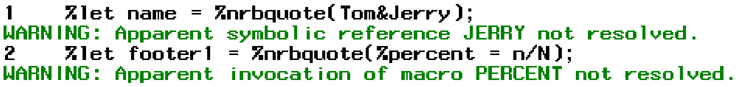
WHEN TO USE A MACRO QUOTING FUNCTION

There are 2 categories for the macro quoting functions. The first is compilation functions including %STR/%NRSTR. It applies to open code such as %LET and the compiled program between %macro statement and %mend statement. In this situation, you need use %STR/%NRSTR for macro quoting, otherwise you may get a warning or error message from SAS log window.

Take blew sample code 1 for example.

```
   %let name = %nrbquote(Tom&Jerry);
   %let footer1 = %nrbquote(%percent = n/N);
```

Log will put warning as blew.



```
1    %let name = %nrbquote(Tom&Jerry);
WARNING: Apparent symbolic reference JERRY not resolved.
2    %let footer1 = %nrbquote(%percent = n/N);
WARNING: Apparent invocation of macro PERCENT not resolved.
```

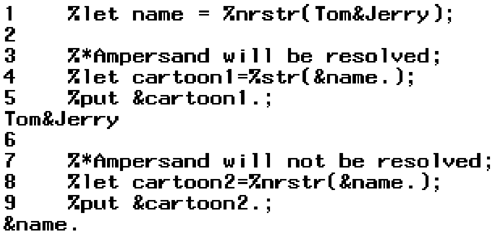
How is macro quoting functions working? During the compilation period, the word scanner will check the code and translate them into a low-level pseudo code language. The macro processor masks special characters by adding prefix and suffix to the string with a hexadecimal character, which called delta character.
The %QUOTE/%NRQUOTE/%BQUOTE/%NRBQUOTE will be treated as normal text in the compilation period. So the special characters will cause some ambiguities.

The other type is execution functions. In the execution period, the SAS processor executes compiled macro program instructions. This type includes %QUOTE/%NRQUOTE/%BQUOTE/%NRBQUOTE. They make the macro processor to treat special characters that result from resolving a macro expression as text. The %STR/%NRSTR also won't mask special characters during this process.

DO YOU NEED AMPERSANDS AND PERCENT SIGNS BE RESOLVED?

If you don't' want the macro triggers (& or %) to resolve a macro variable or call a macro, then you should use the functions which use "NR" for prefix. Here "NR" means "Not Resolved".

The sample code 2 blew will show the difference between "Resolved" and "Not Resolved". In the first part, the macro variable "name" will be resolved to "Tom&Jerry" defined before. In the second part, it is not resolved and put as text directly.



```
1     %let name = %nrstr(Tom&Jerry);
2
3     %*Ampersand will be resolved;
4     %let cartoon1=%str(&name.);
5     %put &cartoon1.;
Tom&Jerry
6
7     %*Ampersand will not be resolved;
8     %let cartoon2=%nrstr(&name.);
9     %put &cartoon2.;
&name.
```

In summaries, you can use the following Figure 1 to help you do the decision.

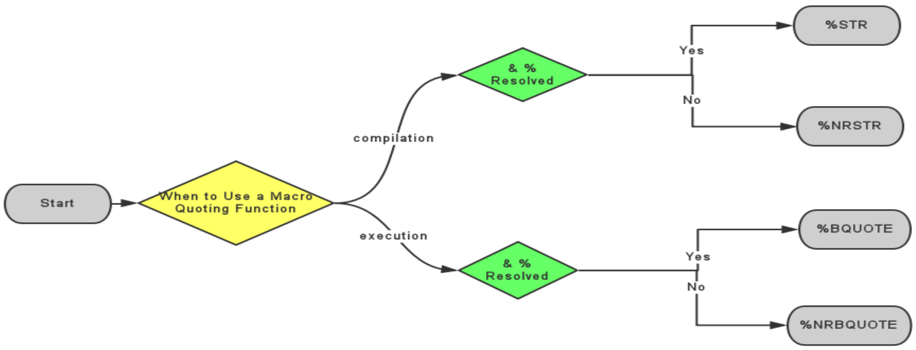


Figure 1 Decision Process to Find Proper Function

We don't choose %QUOTE and %NRQUOTE here because they can be replaced by %BQUOTE and %NRBQUOTE totally without using the preceding percent sign (%).

SPECIAL USAGE

Besides the normal usage of macro quoting functions above, there are some especial situations we need take care of. Here are some examples.

WHERE TO USE EXECUTION FUNCTIONS?

Although the SAS does not specify where you could use the execution functions to resolve macro variable or recall macro. You can either use them inside a macro for a macro parameter or in the invoking step for assigning the parameter. I suggest using them inside a macro in case of any unexpected value for good programming practice. Sometimes, it is not enough. You also need add extra functions to deal with some special situations. In the sample code 3, it will show some situations.

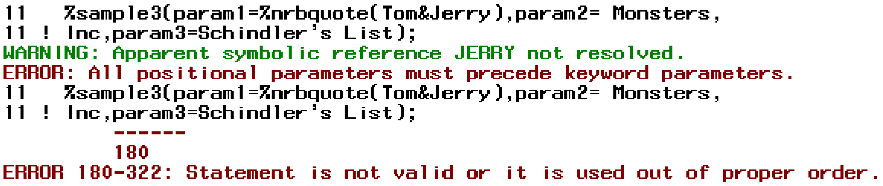
```
%macro sample3(param1=,param2=,param3=);
    %if %bquote(&param1.) ne %then %put
          param1 is valid;
    %if %bquote(&param2.) ne %then %put
          param2 is valid;
    %if %bquote(&param3.) ne %then %put
          param3 is valid;
%mend sample3;

%*parameter lists a;
%sample3(param1=%nrbquote(Tom&Jerry),param2= Monsters,  Inc,param3=Schindler's List);

%* use blew code to jump the loop;
;*';*";*/;quit;run;

%*parameter lists b;
%sample3(param1=%nrstr(Tom&Jerry),param2=%bquote( Monsters,
Inc),param3=%bquote(Schindler's List));
```

For the lists a, "Tom&Jerry" is set to param1 which is not compiled before, so will cause a warning. The part after comma in param2 will add a parameter which is not defined in macro statement and this leads an error. In the last parameter-param3, the single quotation sign will mask the test after itself causing unexpected results. The log is shown as blew.



```
11   %sample3(param1=%nrbquote(Tom&Jerry),param2= Monsters,
11 ! Inc,param3=Schindler's List);
WARNING: Apparent symbolic reference JERRY not resolved.
ERROR: All positional parameters must precede keyword parameters.
11   %sample3(param1=%nrbquote(Tom&Jerry),param2= Monsters,
11 ! Inc,param3=Schindler's List);
         ------
         180
ERROR 180-322: Statement is not valid or it is used out of proper order.
```

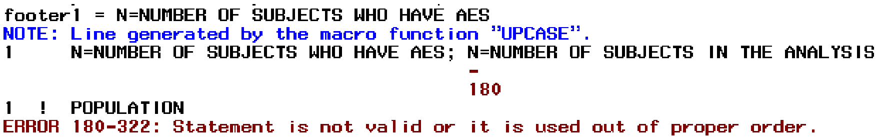
After being changed to the list b, everything is run correctly.

UNEXPECTED UNQUOTING

The special characters under macro quoting functions will be masked until encounter a %UNQUOTE function or passed to next DATA step compiler, PROC step, SAS macro facility or other part of SAS system. However, it is unquoted when you use the %SCAN/%SUBSTR/%UPCASE. This may cause unexpected issue. Please see blew sample code 4 for an example.

```
%let footer1 = %str(n=number of subjects who have AEs; N=number of subjects in the
analysis population);
%put footer1 = %upcase(&footer1.);
```

The log will show an error caused by the first semicolon.



```
footer1 = N=NUMBER OF SUBJECTS WHO HAVE AES
NOTE: Line generated by the macro function "UPCASE".
1      N=NUMBER OF SUBJECTS WHO HAVE AES; N=NUMBER OF SUBJECTS IN THE ANALYSIS
                                         -
                                         180
1   !  POPULATION
ERROR 180-322: Statement is not valid or it is used out of proper order.
```

You can use %QSCAN/%QSUBSTR/%QUPCASE instead of the above functions.

USING %SUPERQ TO AVOID WARNING

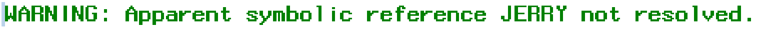
Maybe you have noticed that we haven't mentioned about the %SUPERQ function. It is some kind like %NRBQUOTE. It quotes the first level value of that macro variable in itself without permitting any resolution to occur. It does not mean the variable is not resolved. It means that nothing in the value of that variable will be resolved. You should omit the leading ampersand when using this function. The argument of %SUPERQ is assumed to be the name of a macro variable. This means that if the argument contains an ampersand it will be resolved first and the result is assumed to be the name of a macro variable.

```
DATA null;
    call symput("name","Tom&Jerry");
run;

%*Part A - With warning in log;
%put name = &name.;

%*Part B - Without warning in log by using %superq;
%let newname = %superq(name);
%put name = &newname.;
```

In the part A, the macro "&name" is resolved into "Tom&Jerry" first. Then the processer attempts to resolve the "&Jerry". But it is not defined before. So it will put warning in log as blew.



```
WARNING: Apparent symbolic reference JERRY not resolved.
```

In the part B, the macro name will be translated into "Tom&Jerry" then treated as text forever. The value "Tom&Jerry" will never be resolved to deeper level.

CONCLUSION

Although we have 7 macro quoting functions, %STR/%NRSTR/%BQUOTE/%NRBQUOTE can deal with most of purpose.

If you want to mask the special characters from program to compiler, then using the %STR/%NRSTR functions.

Otherwise from compiler to executable code, using %BQUOTE/%NRBQUOTE.

If you want the trigger will not be resolved, using the "NR" prefixed functions.

REFERENCES

Rosenbloom, Mary F. O.; Carpenter, Art. 2013. "Macro Quoting to the Rescue: Passing Special Characters." *Proceedings of the SAS Global Forum 2013*. San Francisco, California.

O'Connor, Susan. 2003. "Secrets of Macro Quoting Functions – How and Why." Proceedings of the Twelfth Annual Northeast SAS User Group Conference. Washington, DC.

Whitlock, Ian. 2003. "A Serious Look at Macro Quoting." Proceedings of the Twenty-Eighth Annual SAS User Group International Conference, Seattle. WA.

Patterson, Brian; Remigio, Mylene. 2007. "Don't %QUOTE() Me on This: A Practical Guide to Macro Quoting Functions" *Proceedings of the SAS Global Forum 2007*. Orlando, Florida

CONTACT INFORMATION

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

Name: Pengfei Guo
Enterprise: MSD R&D (China) Co., Ltd.
Address: Building A, Headquarters Park, Phase 2, 1582 Gumei Road, Xuhui District
City, State ZIP: Shanghai, 200233
Work Phone: +86 21 2211 8543
Fax: +86 21 2211 8899
E-mail: peng.fei.guo1@merck.com