Paper 86

A MACRO TOOL TO SEARCH AND REPLACE PORTIONS OF TEXT

Jennifer Lin, Pacific Research Associates, Inc., Mountain View, CA

ABSTRACT

This paper describes a macro tool that searches character variables for specified text strings and replaces them with other text strings. This macro is especially useful if the search string occurs multiple times within different character values. For example, if you simply wanted to replace "ABD." with "ABDOMINAL", and you knew that in every instance the character value only contained one word, you could use a simple "if" statement to perform the search and replace (i.e., if text = "ABD." then newtext = "ABDOMINAL"). However, if "ABD." was present in multiple contexts – for instance, “ABD. CRAMPING", "ABD. ASCITES", "ABD. PAIN" -- then rather than identifying and creating a separate "if" statement for each scenario, it would be far simpler to use this macro tool. In the process of explaining the macro code, I will review the following SAS® functions: INDEX, UPCASE, LENGTH, and SUBSTR.

INTRODUCTION

Searching and replacing text within variable values in SAS is not as easy as you might imagine. No function exists that will accomplish a search-and-replace in one easy step. The INDEX function allows you to search for text phrases within values, but does not assist with replacing them. The TRANSLATE function searches and replaces characters within values on a one-on-one basis. It is not helpful if you are trying to replace an entire word or phrase with another word or phrase.

If only a few unique variable values need to be replaced, then the command is simple. For example:

```
if var = ‘ABD.’ then var = ‘ABDOMINAL’;
```

If, however, we realize that the word ‘ABD.’ exists within over, say, 100 different word combinations, we may balk at typing a new SAS statement for every unique replacement.

In these instances, the macro described below can help reduce programming time and errors. ‘ABD.’ can be replaced with ‘ABDOMINAL’ in one quick statement, and a list of all changes is automatically generated for your review. In addition, the macro not only searches and replaces text, but can also create new variables from existing variables.

SOURCE CODE

The complete source code for the search-and-replace macro is provided below:

```
%macro replace (dataset = , chkvar = , search = , replace =,
newvar = );

data &dataset (drop = textpos) newvars;
  length &newvar $ 200;
   set &dataset;

textpos = index(upcase(&chkvar), upcase("&search"));

oldvar = &chkvar;

if textpos ne 0 then
  do;
       if textpos ne 1 then
         &newvar = substr(&chkvar, 1, (textpos - 1))
              ||"&replace"||substr(&chkvar, (textpos +
              length("&search")));
      else &newvar = "&replace"||
             substr(&chkvar, (textpos + length("&search")));
      output newvars;
  end;
else &newvar = &chkvar;

output &dataset;
run;

proc print data = newvars;
  title “SEARCH FOR ‘&search’ AND REPLACE WITH ‘&replace’”;
  var oldvar &newvar;
run;
%mend replace;
```

REVIEW OF FUNCTIONS USED

The macro is coded using four functions – INDEX, UPCASE, LENGTH, and SUBSTR, which are reviewed below:

INDEX

The INDEX function outputs the starting position of a text string within a variable value. If the string is not found in the variable, the position will be assigned a zero value.

For example, index(‘KANGAROO’,’ROO’) would yield a value of 6 because the text string ‘ROO’ begins at the 6th position of the text string ‘KANGAROO’. Index(‘KANGAROO’, ‘TIGER’) would yield a value of 0 because the text string ‘TIGER’ does not appear within the text string ‘KANGAROO’.

UPCASE

The UPCASE function simply reads in a character value and returns the value in upper case letters:

```
UPCASE('Kangaroo') → 'KANGAROO'
```

SUBSTR

The SUBSTR function reads in a character variable, returning the portion of the variable delimited by user-defined positions A and B, inclusive.

For example, if we wanted to obtain the 'KANG' portion of 'KANGAROO', we would instruct SAS to return the substring that begins with the letter in the 1st position and ends with the letter in the 4th position:

```
substr('KANGAROO', 1, 4) → 'KANG'
```

If we wanted to obtain the 'ROO' portion of 'KANGAROO', we would instruct SAS to return the substring that begins with the letter in the 6th position and ends with the letter in the last position. If position B is not defined by the user, SAS assumes that position B is at the end of the variable value:

```
substr('KANGAROO', 6) → 'ROO'
```

LENGTH

The LENGTH function returns the position of the rightmost nonblank character in the argument (i.e., the length of an input phrase). For instance, the value of length('KANGAROO') is 8. Spaces in the middle of the argument are counted, but spaces at the end are not. Thus, the value of length('KANGA ROO') is 9.

EXPLANATION OF MACRO SOURCE CODE

To illustrate how the INDEX, UPCASE, SUBSTR, and LENGTH functions are used to perform a search-and-replace, let us assume we have a data set called AEDATA. Within this data set, there is a variable AETEXT, and we want to replace 'ABD.' with 'ABDOMINAL' where AETEXT = 'ASCITES ABD. - SEVERE'.

Defining Macro Parameters

The macro has five parameters that must be provided by the programmer when the macro is called:

(1) dataset = the name of the data set that contains the character variable that will be searched (AEDATA)
(2) chkvar = the name of the character variable that will be searched (AETEXT)
(3) search = the search string ('ABD.')
(4) replace = the text string that will replace the search string ('ABDOMINAL')
(5) newvar = the name of the new character variable containing the replace string (AETEXT)

Note that in a search-and-replace, the CHKVAR and NEWVAR parameters should be the same. If you are creating a new variable from an existing variable, set CHKVAR equal to the existing variable and NEWVAR equal to the new variable.

Setting the Length of NEWVAR

The length of the new variable is set to 200 bytes – the maximum allowed in SAS version 6 – in the following statement:

```
length &newvar $ 200;
```

If the LENGTH statement is not used, the new variable may be truncated.

When performing a search-and-replace, the length defaults to the length defined for the original variable. If the replace string is longer than the search string, the variable after replacement will be longer than the variable prior to replacement, and the likelihood of truncation increases.

When creating a new variable, the length will default to the length of the first value assigned to the variable. For example, if the first value assigned to the new variable is 'ABDOMINAL PAIN', the remaining values are truncated at 14 bytes.

Performing the Search-and-Replace

The INDEX function is used to search the check variable for the search string and store the position of the first letter of the search string in the variable TEXTPOS:

```
textpos = index(upcase(&chkvar), upcase("&search")); →
textpos = index(upcase(aetext),upcase("ABD.")); →
textpos = index(upcase('ASCITES ABD. – SEVERE')
            ,upcase("ABD.")); →
textpos = 9;
```

Note that the UPCASE function is optional; you can use it if you do not want the search to be case-sensitive.

Because the first letter of the search string 'ABD.' Is in the 9th position of the phrase 'ASCITES ABD. – SEVERE', TEXTPOS = 9.

To obtain the text that occurs before the text string, use:

```
substr(&chkvar, 1, (textpos - 1)) →
substr(aetext, 1, (9 – 1)) →
substr(‘ASCITES ABD. – SEVERE’, 1, 8) →
‘ASCITES’
```

The SUBSTR function above returns the value ‘ASCITES ‘.

To obtain the text that occurs after the text string, use:

```
substr(&chkvar, (textpos + length("&search"))) →
substr(aetext, (9 + length(‘ABD.))) →
substr(‘ASCITES ABD. – SEVERE’, (9 + 4)) →
substr(‘ASCITES ABD. – SEVERE’, 13) →
‘ – SEVERE’
```

The length of the search string ‘ABD.’ is 4. Thus, SAS returns the substring which begins in the 13th position and ends in the last position: ‘ - SEVERE’;

Because the search string is present and does not occur at beginning of the variable (i.e., TEXTPOS does not equal 0 or 1), the replace string is inserted between the text that occurs before the search string and the text that occurs after the search string:

```
&newvar →
text before search string||replace string||text after search string →
‘ASCITES ‘||’ABDOMINAL’||’ - SEVERE’ →
‘ASCITES ABDOMINAL SEVERE’
```

If the search string occurs at the beginning of variable (e.g., ‘ABD.’ in ‘ABD. CRAMPING’), we require different code. Otherwise, TEXTPOS = 1 so the SUBSTR function will produce an error because TEXTPOS - 1 = 0. This code simply concatenates the replace string with the text that occurs after the search string:

```
"&replace"||substr(&chkvar, (textpos + length("&search"))) →
replace string||text after search string →
‘ABDOMINAL’||’ CRAMPING’ →
‘ABDOMINAL CRAMPING’
```

If the search string is not present, then the old variable is assigned to the new variable name without any changes:

```
else &newvar = &chkvar;
```

Reviewing the Output

Notice that we output all the changed variables into a separate data set called NEWVARS, which is then printed for review.

It is very important to always check the .lst output to make sure that text was replaced correctly. Otherwise, some unfortunate errors may slip by unnoticed. For instance, if we accidentally searched on the letters "ABD" instead of "ABD.", the word "ABDOMINAL" would be replaced to read "ABDOMINALOMINAL".

EXAMPLE OF MACRO USE

This sample data set contains patient identification numbers and adverse events:

```
OBS     PTID    AETEXT

1       110     ABD. CRAMPING
2       160     INTRA-ABD. ABSCESS
3       190     (L) ABD. PAIN
4       220     FOOT ULCER (L) FOOT
5       230     ABD. PAIN
6       360     (L) ELBOW CONTUSION
7       440     PAIN (L) KNEE
8       520     ACNE
9       520     TINGLING (L) FOOT
10      530     ASCITES ABD.
```

Macro-Less Search-and-Replace

First identify all the variable values that must be replaced:

```
if index(aetext, ‘ABD.’) ne 0 or
   index(aetext, ‘(L’) ne 0 then output replace;

proc print data = replace;
```

This code produces the following output:

```
 OBS    PTID    AETEXT

1       110     ABD. CRAMPING
2       160     INTRA-ABD. ABSCESS
3       190     (L) ABD. PAIN
4       220     FOOT ULCER (L) FOOT
5       230     ABD. PAIN
6       360     (L) ELBOW CONTUSION
7       440     PAIN (L) KNEE
9       520     TINGLING (L) FOOT
10      530     ASCITES ABD.
```

Next, instruct SAS to replace each of these values with the new values:

```
If aetext = ‘ABD. CRAMPING’ then
  aetext = ‘ABDOMINAL CRAMPING’;
If aetext = ‘INTRA-ABD. ABSCESS’ then
  aetext = ‘INTRA-ABDOMINAL ABSCESS’;
If aetext = ‘(L) ABD. PAIN’ then
  aetext = ‘LEFT ABDOMINAL PAIN’;
If aetext = ‘FOOT ULCER (L) FOOT’ then
  aetext = ‘FOOT ULCER LEFT FOOT’;
If aetext = ‘ABD. PAIN’ then
  aetext = ‘ABDOMINAL PAIN’;
If aetext = ‘(L) ELBOW CONTUSION’ then
  aetext = ‘LEFT ELBOW CONTUSION’;
If aetext = ‘PAIN (L) KNEE’ then
  aetext = ‘PAIN LEFT KNEE’;
If aetext = ‘TINGLING (L) FOOT’ then
  aetext = ‘TINGLING LEFT FOOT’;
If aetext = ‘ASCITES ABD.’ then
  aetext = ‘ASCITES ABD.’;
```

Note that in this example, there are only 9 replacements. Obviously, the macro can save even more time when the number of replacements is greater.

Furthermore, additional lines of code are required if you want to produce output to check that all replacements were correctly performed.

Using the Search-and-Replace Macro

Ideally the search-and-replace macro should be saved in a separate "tools" directory, which can be accessed from any study. It can then be inserted into your program using the %INCLUDE statement:

```
%include "../tools/replace.sas";
```

Within the program, and outside of the DATA step, call the macro:

```
%replace (dataset = aes,
          chkvar = aetext,
          search = ABD.,
          replace = ABDOMINAL,
          newvar = aetext);
%replace (dataset = aes,
          chkvar = aetext,
          search = (L),
          replace = LEFT,
          newvar = aetext);
```

Review the .lst output to make sure each change was correctly performed. The .lst output looks like this:

```
SEARCH FOR 'ABD.' AND REPLACE WITH 'ABDOMINAL'

OBS  OLDVAR                AETEXT

1    (L) ABD. PAIN         (L) ABDOMINAL PAIN
2    ABD. CRAMPING         ABDOMINAL CRAMPING
3    ABD. PAIN             ABDOMINAL PAIN
4    INTRA-ABD. ABSCESS    INTRA-ABDOMINAL ABSCESS
5    ASCITES ABD.          ASCITES ABDOMINAL

SEARCH FOR '(L)' AND REPLACE WITH 'LEFT'

OBS  OLDVAR                AETEXT

1  (L) ABDOMINAL PAIN     LEFT ABDOMINAL PAIN
2  (L) ELBOW CONTUSION    LEFT ELBOW CONTUSION
3  FOOT ULCER (L) FOOT    FOOT ULCER LEFT FOOT
4  TINGLING (L) FOOT      TINGLING LEFT FOOT
5  PAIN (L) KNEE          PAIN LEFT KNEE
```

Final Data Set

For both the macro-less search-and-replace and the search-and-replace using the macro, the final data set looks like this:

```
OBS    PTID    AETEXT

1      110     ABDOMINAL CRAMPING
2      160     INTRA-ABDOMINAL ABSCESS
3      190     LEFT ABDOMINAL PAIN
4      220     FOOT ULCER LEFT FOOT
5      230     ABDOMINAL PAIN
6      360     LEFT ELBOW CONTUSION
7      440     PAIN LEFT KNEE
8      520     ACNE
9      520     TINGLING LEFT FOOT
10     530     ASCITES ABDOMINAL
```

CONCLUSION

The search-and-replace macro is a handy tool to add to your tool kit. It can both perform a search-and-replace and create a new variable from an existing variable. The code is short and easily understandable, and the macro itself is user-friendly. Used properly, it should decrease your programming time and minimize the likelihood of error.

ABOUT THE AUTHOR

Jennifer was introduced to SAS programming while pursuing a Master's Degree in Epidemiology at the University of Michigan. Upon graduation, she held a research internship at the Centers for Disease Control and Prevention in Atlanta. She is now a Programmer/Analyst at Pacific Research Associates, a contract research organization in Mountain View, California. Pacific performs research for biotech, pharmaceutical, and medical device companies.

CONTACT INFORMATION

Questions and comments are welcome at:

Jennifer Lin
Pacific Research Associates, Inc.
2570 West El Camino, Fourth Floor
Mountain View, CA 94040
WORK: 650-917-3665
FAX: 650-917-3683
Email: jennifer@prai.com