PharmaSUG 2014 - Paper CC38

Let SAS® Do That For You

Emmy Pahmer, inVentiv Health Clinical, Montreal, Canada

ABSTRACT

Do you ever repeat certain tasks and think there should be a better or easier way to do them? Choosing and copying files are tasks that we often perform manually but which can easily be done by SAS using particular selection criteria. We'll look at how to get a list of files in a directory, some code to select the desired file, creating a warning message if more than one file fits the criteria or if no files fit the criteria, and copying the file(s). This presentation is suitable for all users, including beginners.

INTRODUCTION

We don't normally think of using SAS for tasks "outside" SAS, but this example will demonstrate that you can make life easier for yourself by letting SAS do some of them for you. In this example, we will look at the contents of a directory, choose a file and copy it to another location.

FILENAME STATEMENT

First we identify the directory to be checked with a FILENAME statement. Using the operating system command DIR, a list of the directory contents is generated. With the PIPE option, the output is sent to the designated file:

```
filename speclist pipe  'dir "R:\ABC123\documents" ';
```

Notice the single quotes around the double quotes. This makes resolving macro variables within them a little tricky. One way of doing this is by using this little macro:

```
%macro single(v);
    %unquote(%str(%')&v%str(%'))
%mend single;

%let study = ABC123;
filename speclist pipe  %single(dir "R:\&study\Documents");
```

Here is the same statement as above, but using a macro variable and the SINGLE macro to resolve it. The macro variable is resolved first and then the single quotes are put around the DIR statement.

INTO SAS

Create a SAS dataset, using the filename to input the contents of the directory. Here we are putting each line into a variable called BUFFER.

```
data speclist1 ;
    length buffer $256;
    infile speclist length=reclen ;
    input buffer $varying256. reclen ;
run;

proc print data = speclist1 ; run;
```

Here's what the contents look like with PROC PRINT:

```
Obs    buffer
  1  Volume in drive R has no label.
  2  Volume Serial Number is 58C5-EACA
  3
  4  Directory of R:\Affaires Scientifiques\Data Management\Program\SAS
Conferences\pharmasug2014_EVP\
  5
  6 2014-04-08  22:35    <DIR>          .
  7 2014-04-08  22:35    <DIR>          ..
  8 2014-04-08  16:16            72 704 ABC123_Database_Specifications_Draft02.xls
  9 2014-04-08  16:16            72 704 ABC123_Database_Specifications_Final02.xls
 10 2014-04-08  22:35            72 192
ABC123_Database_Specifications_Final03_ongoing.xls
 11 2014-04-08  16:17    <DIR>          Old
 12                3 File(s)        217 600 bytes
 13                3 Dir(s)  35 315 838 976 bytes free
```

Output 1. PROC PRINT output of SPECLIST1

CODING THE CRITERIA

Normally, outside of SAS, we would be looking at this directory, possibly with Windows Explorer, and using certain criteria to decide which file is the current one, and therefore the one to copy. We'll recreate the logic using SAS to choose the correct file.

```
data speclist2 (keep = file_name date time ongoing final);         * (08) *;
    set speclist1;
    ubuffer = compbl(upcase(buffer));
    if find(ubuffer, 'XLS') = 0  then delete;                      * (01) *;
    date = scan(ubuffer, 1, ' ');                                  * (02) *;
    time = scan(ubuffer, 2, ' ');
    length  file_name $200;                                        * (03) *;
    nwords = count(strip(ubuffer), ' ') + 1;
    do i = 1 to nwords;                                            * (04) *;
        word = scan(ubuffer, i, ' ');
        if index(upcase(word), 'XLS') then file_name = word;
    end;
    if index(file_name, 'SPEC') then
    do;
        if index(file_name, 'ONGOING') then ongoing = 'Y' ;        * (05) *;
        else if index(file_name, 'FINAL') then final = 'Y';        * (06) *;
        if final = 'Y' or ongoing = 'Y' then output;               * (07) *;
    end;
run;
```

01. Keep only records that contain "XLS"
02. Create variables for date and time
03. Create a variable for the file names
04. Loop through each token (or word) in the record to find the one with the XLS (or XLSX) extension. Depending on the operating system, this may change location (ie, not always the 5th token)
05. Flag the "ongoing" ones.
06. Flag the "final" ones.
07. Output if either ongoing or final.
08. Keep only date, time, file_name and two flags.

Here are the contents of this new dataset:

```
Obs       date       time                          file_name
1     2014-04-08    16:16    ABC123_DATABASE_SPECIFICATIONS_FINAL02.XLS
2     2014-04-08    22:35    ABC123_DATABASE_SPECIFICATIONS_FINAL03_ONGOING.XLS
```

Output 2. PROC PRINT output of SPECLIST2

Normally there would only be one file here, since older versions should be moved to a different sub-directory. Now we could just take the more recent one of the two, examining the date/times, but in this case we will use different criteria.

```
%global currentspec;

data speclist3;
    set speclist2 end = at_end;
    retain n_ongoing 0 n_final 0;
    if ongoing = 'Y' then n_ongoing + 1;                    * (11) *;
    if final = 'Y' then n_final + 1;                        * (12) *;
    if at_end then
    do;
        if n_final > 0 and n_ongoing > 0 then put 'WARNING: Both ongoing and final are
present.';                                                  * (13) *;
        else if n_ongoing > 1           then put 'WARNING: More than one ONGOING spec
file.';
        else if n_final > 1             then put 'WARNING: More than one FINAL file
found.';
        put n_final= n_ongoing=;
        if n_final = 1 or n_ongoing = 1 then
        do;
            put 'Current spec file is ' file_name 'dated ' date time;
            call symputx('currentspec', file_name);         * (14) *;
        end;
    end;
run;
```

11. Count how many files are ongoing.
12. Count how many files are final.
13. There should only be one so we'll issue warnings if there are too many or old files seem to be present. In that case, old files would need to be moved.
14. Populate the macro variable with the file name.

COPYING THE FILE

One way of copying the file is by using the %SYSEXEC statement, which executes operating system commands.

```
%sysexec copy "R:\&study\Documents\&currentspec"
               "&root\Data\Input\&currentspec";
```

One can also use the X command. There are several discussions and papers available on this topic.

CONCLUSION

Any time we perform repetitive tasks there is probably a way to automate them and often, they can easily be done with SAS, even if they're normally performed by the operating system.

REFERENCES

FILENAME Statement: Windows
http://support.sas.com/documentation/cdl/en/hostwin/63285/HTML/default/viewer.htm#chfnoptfmain.htm

Using Unnamed Pipes
http://support.sas.com/documentation/cdl/en/hostwin/63285/HTML/default/viewer.htm#unnamed.htm

Quick 'n Dirty - Small, Useful Utility Macros - Harry Droogendyk
http://www.stratia.ca/papers/utility_macros.pdf

%SYSEXEC Statement
http://support.sas.com/documentation/cdl/en/mcrolref/61885/HTML/default/viewer.htm#a000171045.htm

CONTACT INFORMATION

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

Emmy Pahmer
inVentiv Health Clinical
Montreal, Canada
emmy.pahmer@inventivhealth.com