PharmaSUG 2019 - Paper AP-074

Where Is the Information We Are Looking For?

Jeff Xia, Merck & Co., Inc., Rahway, NJ, USA;

Shunbing Zhao, Merck & Co., Inc., Rahway, NJ, USA

ABSTRACT

Finding historical analysis and reporting artifacts in an efficient and effective way is an essential need for a statistical programmer. Identifying the source data, SAS programs or macros used to create a previously produced analysis, table, or figure can be challenging especially when facing a tight timeline and on old projects where institutional or historical knowledge may be limited. Statistical programmers need tools to trace back and search for these historical artifacts and effectively locate various inputs for analysis and reporting deliverables.

This paper presents four innovative utilities to programmatically find the information of interest. 1) Finding a text string in all text files, including text files in subdirectories. This utility is extremely useful when there is a need to locate a certain CSR table out of hundreds tables; it is also useful in understanding which SAS program was used to generate a specific table, which macros were involved, etc. 2) Finding all records and fields that contain certain text string in all the datasets in a SAS library, it becomes handy when we need to understand the SDTM mapping from raw datasets, or from SDTM to ADaM, etc. 3) Finding a variable name in all datasets in a SAS library, it is useful when working on legacy studies in which variables do not follow CDISC standard naming convention. 4) Finding files of interest in a directory and/or its subdirectories, the file name can be partial, or in a pattern. These four utilities have been widely used in our daily programming activities, which significantly improved our capability in working on time-sensitive tasks.

INTRODUCTION

Responding efficiently to information requests about analysis and reporting deliverables is needed. Information requests can be received from internal company stakeholders, i.e., statisticians, clinical, regulatory colleagues, and from external stakeholders including regulatory agencies and investigators. Responding effectively can be challenging for a statistical programmer especially when doing so on an historical study. Additionally, staffing changes, competing priorities, and timelines can further complicate the task. This paper presents four utilities that can be used to help programmers find the key information of interest in an efficient and effective way.

UTILITY 1: FIND FILES CONTAINING A SPECIFIC TEXT STRING

There are many occasions in which we would need to perform a thorough search for a specific text in all the text files, which could be in a main directory or its subdirectories. It is extremely inefficient to open each text file and look for the text of interest manually. The macro below (find0text0in0textfiles.sas) was developed to perform this search programmatically.

The macro find0text0in0textfiles takes 4 macro parameters: rootPath, searchString, filetype, and debug. A sub-macro, “list0files”, was designed to retrieve the name of a list of files with the extension of “&filetype” in the path of “&rootPath” including all its subdirectories. A recursive call was used inside the macro itself to explore the root folder and all its subdirectories and list every qualified file. See below for the key part of the macro.

```
%* Assigns a fileref to the directory and opens the directory *;
%let rc=%sysfunc(filename(filrf,&dir));
%let did=%sysfunc(dopen(&filrf));

%* Make sure directory can be open *;
%if &did eq 0 %then %do;
```

```
        %put Directory &dir cannot be open or does not exist;
        %return;
    %end;

    %* Loops through entire directory *;
    %do i=1 %to %sysfunc(dnum(&did));
        %* Retrieve name of each file *;
        %let fnm=%qsysfunc(dread(&did,&i));
        %let fid=%qsysfunc(mopen(&did,&fnm));

        %* Checks to see if the extension matches the parameter value *;
        %if %qupcase(%qscan(&fnm,-1,.)) = %upcase(&ext) and
            &fid ne 0 %then %do;
            fileloc="%qsysfunc(finfo(&fid,Filename))";
            output;
        %end;
        %* If directory name call macro again *;
        %else %if %qscan(&fnm,2,.) = %then %do;
            %list0files(&dir&mslash%unquote(&fnm),&ext);
        %end;
    %end;
```

A macro call like the one below captures the full path of all text files of interest in the dataset “files”.

```
data files(keep=fileloc);
    length fileloc $2000;
    %list0files(&rootpath,&filtype);
run;
```

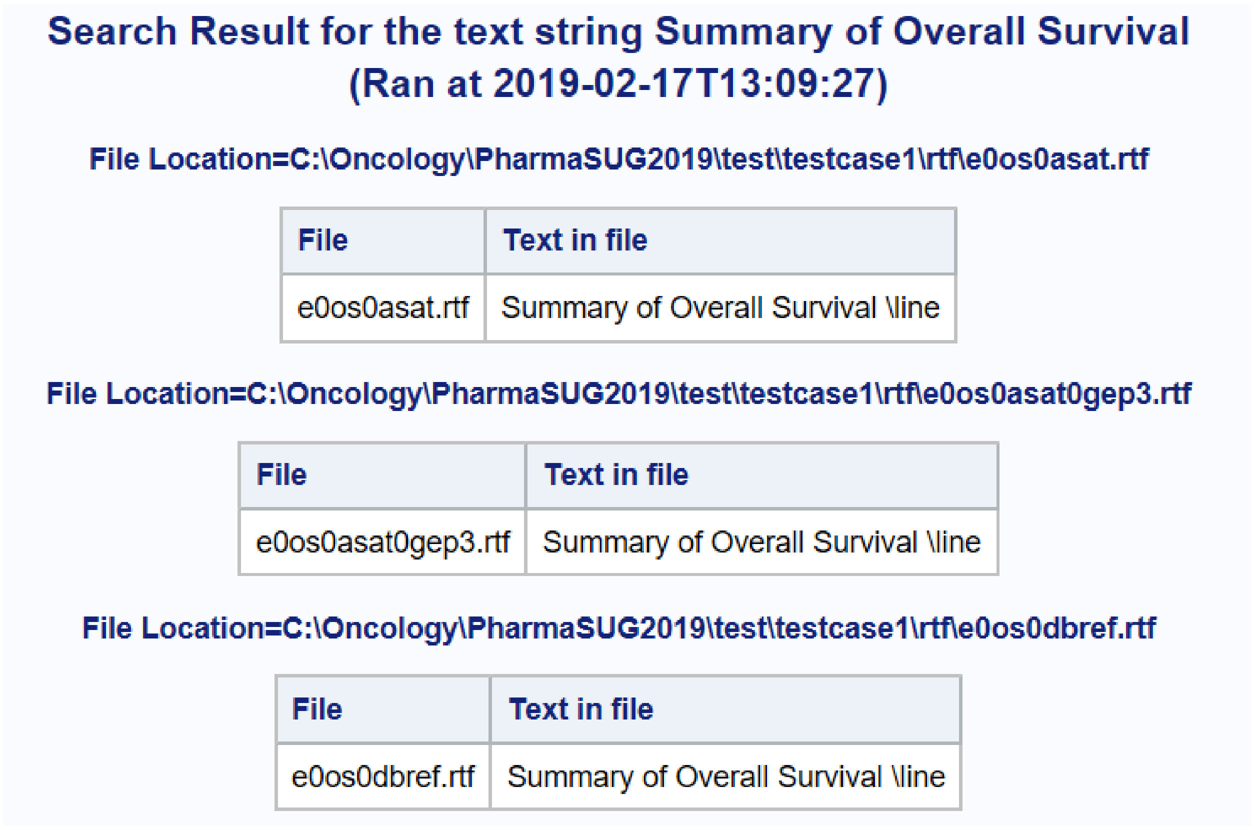
Then, we further process the list of the qualified files and read each text file line by line into another dataset “result”, and only keep the line of text that contains the text of interest “&searchString”, case insensitive.

```
data result(drop = text1);
    length filloc text text1 $2000;
    set files;
    filloc = fileloc;
    do until (eof);
        infile dummy lrecl=5767 end=eof filevar=fileloc;
        input;
        text = strip(_infile_);
        text1 = upcase(_infile_);
        if find(upcase(text1),upcase("&searchString")) then output;
    end;
run;
```

The macro has been proven to be a useful tool in many situations. For example, very often, we receive requests from internal or external stakeholders for additional subgroup analysis on a certain population or a specific age group. We know from the title of the existing output that the stakeholder is interested in, for example, “Summary of Overall Survival”. What is the next step in this situation?

First, we search how many rtf files contains the text “Summary of Overall Survival”. A macro call like the one below serves the purpose.

```
%let text = Summary of Overall Survival;
%find0text0in0textfiles(
         rootPath        = C:\Oncology\PharmaSUG2019\test\testcase1\rtf,
         filType         = rtf,
         searchString    = %str(&text),
         debug           = Y);
```



Search Result for the text string Summary of Overall Survival (Ran at 2019-02-17T13:09:27)

File Location=C:\Oncology\PharmaSUG2019\test\testcase1\rtf\e0os0asat.rtf

File	Text in file
e0os0asat.rtf	Summary of Overall Survival \line

File Location=C:\Oncology\PharmaSUG2019\test\testcase1\rtf\e0os0asat0gep3.rtf

File	Text in file
e0os0asat0gep3.rtf	Summary of Overall Survival \line

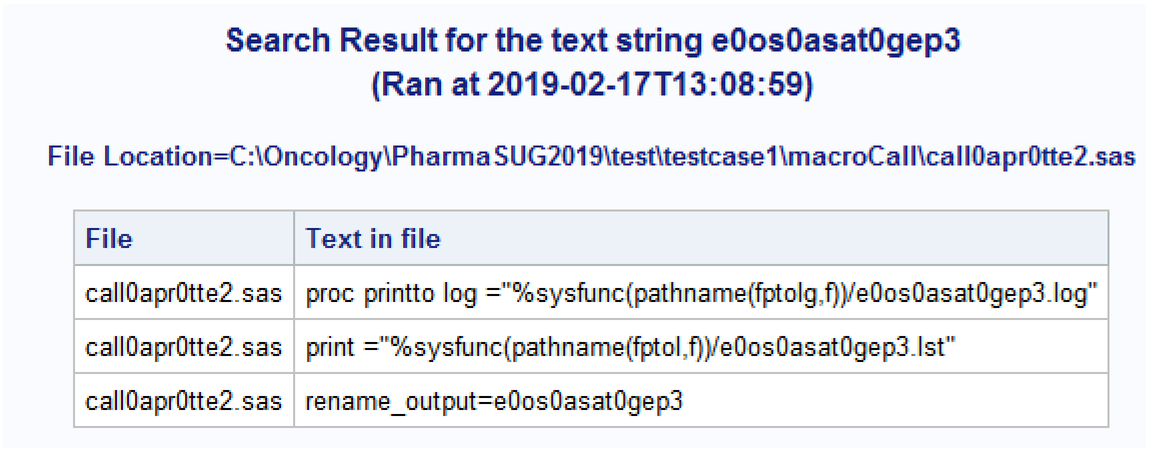
File Location=C:\Oncology\PharmaSUG2019\test\testcase1\rtf\e0os0dbref.rtf

File	Text in file
e0os0dbref.rtf	Summary of Overall Survival \line

Figure 1. Sample output of searching text in RTF files.

Once we know the output rtf name, the next question to ask is: which macro generates this output? A macro call like the one below is the answer to the question.

```
%let text = e0os0asat0gep3;
%find0text0in0textfiles(
      rootPath     = C:\Oncology\PharmaSUG2019\test\testcase1\macroCall,
      filType      = sas,
      searchString = %str(&text),
      debug        = Y);
```



Search Result for the text string e0os0asat0gep3 (Ran at 2019-02-17T13:08:59)

File Location=C:\Oncology\PharmaSUG2019\test\testcase1\macroCall\call0apr0tte2.sas

File	Text in file
call0apr0tte2.sas	proc printto log ="%sysfunc(pathname(fptolg,f))/e0os0asat0gep3.log"
call0apr0tte2.sas	print ="%sysfunc(pathname(fptol,f))/e0os0asat0gep3.lst"
call0apr0tte2.sas	rename_output=e0os0asat0gep3

Figure 2. Sample output of searching text in SAS file.

Sometimes we have global updates to make in macro libraries, including the global macro library, TA standard library, as well as the project specific library. For example, there was a need to convert the day to month to meet specific reporting needs. In the past we were inconsistent in the conversion factors, such as 30.4367 or 30.4375. The difference of the factors could be significant in certain reporting activities, such as DSUR or RMP. A programmatic search like the call below finds every case in the macro library, and makes it convenient when performing a global update.

```
%let text = 30.4375;
%find0text0in0textfiles(
     rootpath       = C:\Oncology\PharmaSUG2019\test\testcase1\macro,
     filtype        = sas,
     searchstring   = %str(&text),
     debug          = Y);
```

UTILITY 2: SEARCH TEXT OF INTEREST IN A SAS LIBRARY

The macro `find0text0in0dataset` takes 7 parameters: rootPath, ds, subj, text, ignoreCase, exact, and debug. A user can specify the location of the datasets to be searched, by default the utility searches all character variables in all datasets and captures all cases of interest. Alternatively, the user can limit the output by specifying a dataset name to search and/or a specific subject to search. The search can be case sensitive or insensitive, an exact match or text of interest (ignoring letter cases).

Firstly, we define a user library with read only access. The list of SAS datasets in that library can be obtained by using the SAS procedure proc contents. Then create a list of local macro variables sas1 to sas`totfile` where `totfile` represents total number of datasets in the library.

```
libname rawdata "&rootPath" access = readonly;
proc contents data = rawdata._all_
              out = myds(keep = memname name)
              noprint;
run;

proc sort data = myds(keep = memname)
          nodupkey; *to obtain a list of dataset name
    by memname;
run;
data files;
    set myds end = eof;
    call symput("sas"||strip(put(_n_,4.)),strip(memname));
    if eof then call symput("totfile", strip(put(_n_,4.)));
run;
```

Next step, a do loop is built to go through each dataset and check whether it contains a variable USUBJID If true, set the value of the macro variable `wherecondition` to 1. Otherwise, set it to 0.

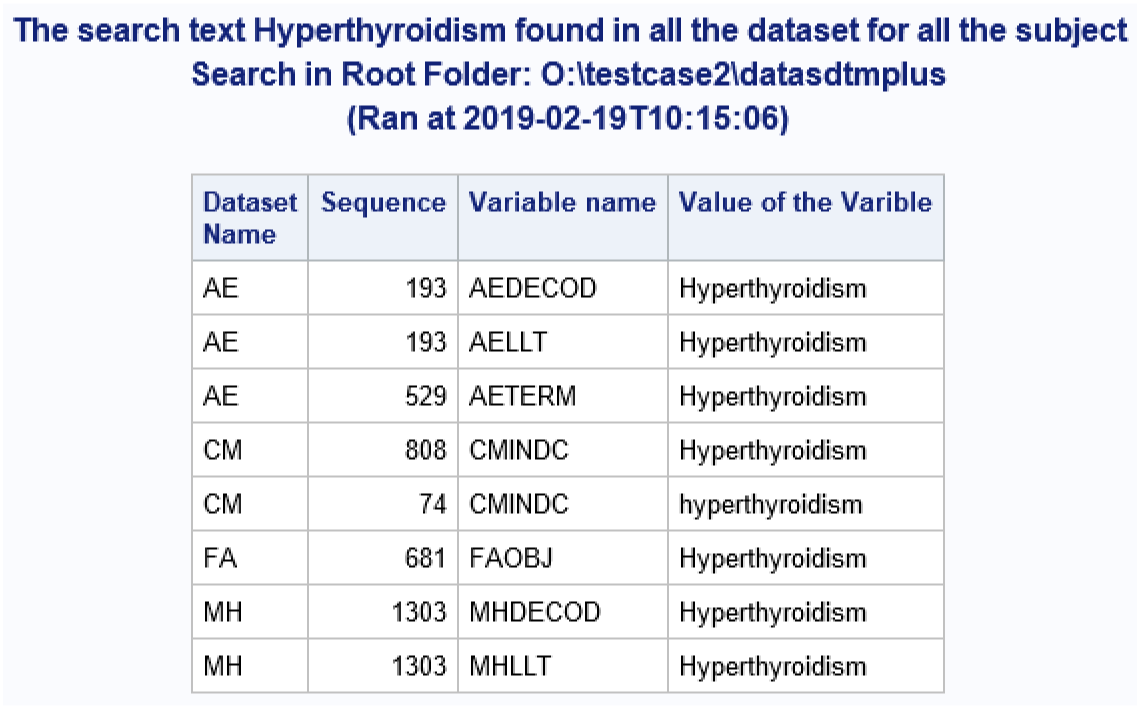
Another loop is designed to go through every character variable in every dataset in the entire SAS library and find all data points that match the text of interest. Before the loop starts, a temporary dataset “_tmp1” is created to save the search result in every iteration of the loop. Then, SAS loops through each record in every dataset in the library and recodes its findings in the dataset “_tmp1”.

```
data _tmp1;
    delete;
run;

data _tmp1(keep = dsn seq varName varValue);
    label seq = 'Observation Number'
          varName = 'Variable Name';
    set rawdata.&&sas&_i;
    %if &whereCondition = 1 %then %do;
        where USUBJID = "&subj";
    %end;
    length dsn $50 varName  $20 varValue $200;
    dsn = "&&&sas&_i";
    array all[*] _character_;
    seq = _N_;
    do i = 1 to dim(all);
        %if &ignoreCase = N %then %do;
            %if &exact = N %then %do;
                if index(all[i], "&text") then do;
            %end;
            %else %if &exact = Y %then %do;
                if strip(all[i]) = strip("&text") then do;
            %end;
        %end;
        %else %do;
            %if &exact = N %then %do;
                if index(upcase(all[i]), upcase("&text")) then do;
            %end;
            %else %if &exact = Y %then %do;
                if strip(upcase(all[i])) = strip(upcase("&text")) then do;
            %end;
        %end;
            varname = vname(all[i]);
            varvalue = all[i];
        *put "find it in dataset &&sas&_i: " seq=   varName = varValue = ;
            if upcase(varName) ne 'VARVALUE' then output;
        end;
    end;
run;
```

This macro is useful when we need to search for a specific text in the database but do not know the name of the variable, or even the name of the dataset. One time, we received a request from the clinical team asking for a specific adverse event for a given subject; it was not captured in the AE panel in the database. We used this utility and searched the entire database for that subject. It turned out that the investigator put some text in the comment field, and the data was mapped to CO domain. DM queried the site based on the search result and asked the site to create a separate AE for that event.

```
%let root = c:\testcase2\datasdtmplus;
%find0text0in0dataset(rootPath   = &root,
                      ds         = all,
                      subj       = ,
                      ignoreCase = Y,
                      exact      = Y,
                      text       = Hyperthyroidism);
```

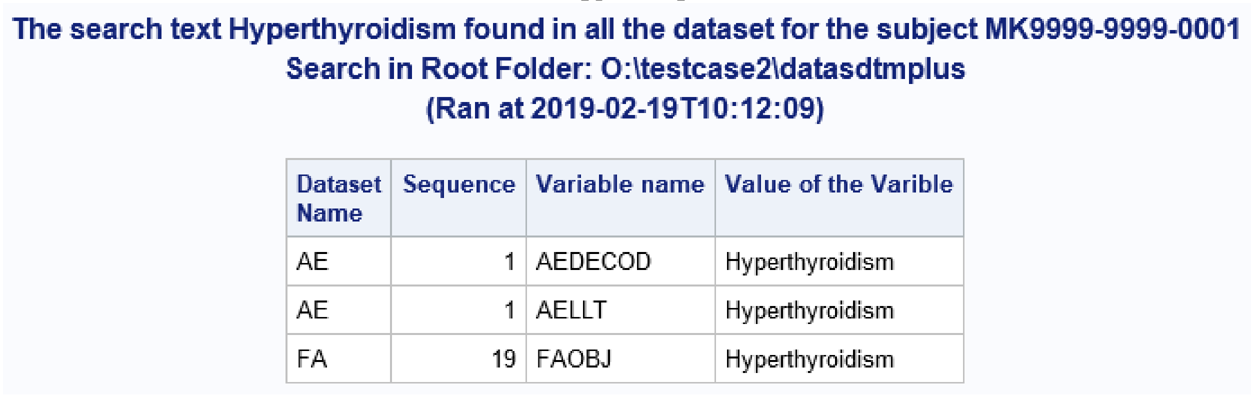

The search text Hyperthyroidism found in all the dataset for all the subject
Search in Root Folder: O:\testcase2\datasdtmplus
(Ran at 2019-02-19T10:15:06)

Dataset Name	Sequence	Variable name	Value of the Varible
AE	193	AEDECOD	Hyperthyroidism
AE	193	AELLT	Hyperthyroidism
AE	529	AETERM	Hyperthyroidism
CM	808	CMINDC	Hyperthyroidism
CM	74	CMINDC	hyperthyroidism
FA	681	FAOBJ	Hyperthyroidism
MH	1303	MHDECOD	Hyperthyroidism
MH	1303	MHLLT	Hyperthyroidism

Figure 3. Sample output of searching text in all SAS datasets for all subjects.

Another good example is to help programmers to understand the data flow in the analysis reporting and submission process, such as the mapping from raw database to SDTM, or from SDTM to ADaM. The macro call below and its output show the search result of the text "`Hyperthyroidism`" for a specific subject "`MK9999-9999-0001`".

```
%find0text0in0dataset(rootPath   = &root,
                      ds         = all ,
                      subj       = MK9999-9999-0001,
                      ignoreCase = Y,
                      exact      = Y,
                      text       = Hyperthyroidism);
```


The search text Hyperthyroidism found in all the dataset for the subject MK9999-9999-0001
Search in Root Folder: O:\testcase2\datasdtmplus
(Ran at 2019-02-19T10:12:09)

Dataset Name	Sequence	Variable name	Value of the Varible
AE	1	AEDECOD	Hyperthyroidism
AE	1	AELLT	Hyperthyroidism
FA	19	FAOBJ	Hyperthyroidism

Figure 4. Sample output of the search text in all SAS datasets for a specific subject.

UTILITY 3: SEARCH A VARIABLE NAME IN THE ENTIRE LIBRARY

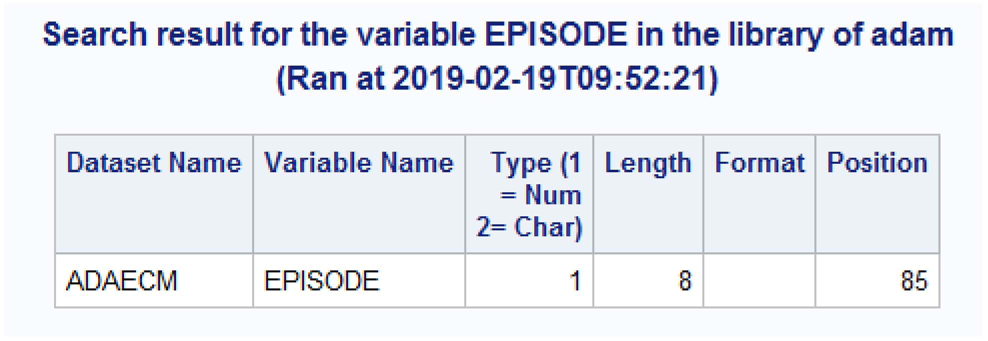
The macro `find0var0in0dataset` takes 4 parameters: lib, var, exactMatch, and debug. It searches in the entire library and sees if any dataset contains a certain variable (&var). If exactMatch = Y, it only captures the records with an exact match. For the code, see below.

```
proc contents data = &lib.._all_
              out = _conts(keep = memname name type length format varnum)
              noprint;
run;

data _conts;
    set _conts;
    %if &exactMatch = Y %then %do;
        if strip(upcase(name)) = upcase(strip("&var"));
    %end;
    %else %do;
        if index(strip(upcase(name)), upcase(strip("&var")));
    %end;
run;
```

The macro is very useful when we work on legacy studies, in which the variable naming conversion does not follow CDISC standard. We used to work on a very old study, the annotated CRF did specify variable names for all data fields, but there was no dataset name specified. It was a struggle for the team to find out the correct dataset name in a plethora of datasets. With this utility, it becomes an easy job to perform. It is also useful for finding variables in ADaM datasets because sometimes it is difficult to know which dataset a variable belongs to by looking at the variable name itself.

```
libname adam 'O:\testcase3\dataanalysis' access = readonly;
%find0var0in0dataset(lib        = adam,
                     var        = EPISODE,
                     exactMatch = Y,
                     debug      = N );
```



Search result for the variable EPISODE in the library of adam
(Ran at 2019-02-19T09:52:21)

Dataset Name	Variable Name	Type (1 = Num 2= Char)	Length	Format	Position
ADAECM	EPISODE	1	8		85

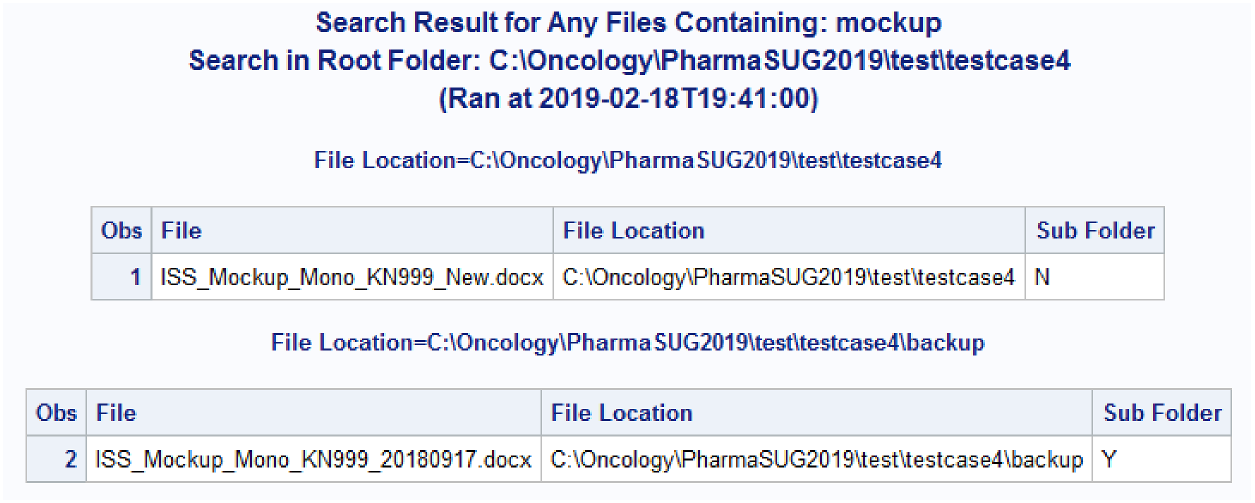
Figure 5. Sample output of searching variable in all SAS datasets in a library.

UTILITY 4: SEARCH A FILENAME IN THE ENTIRE FOLDER INCLUDING SUB FOLDERS

The macro `find0file0in0Folders` takes four parameters: rootPath, searchString, filetype, and debug. First, it uses the macro “list0files” to obtain a list of file names inside the folder of “&rootPath”, the file type can be specified by using the parameter filetype, i.e., if filetype = sas, then the utility searches all SAS program files with the file extension of “sas”.

This macro can be very useful when we know the partial name of a file but do not know the exact name, where it was saved, or where the latest version was saved. A search generates a report with all the information in details.

```
%let text = mockup;
%find0file0in0Folders(
    Rootpath      = C:\Oncology\PharmaSUG2019\test\testcase4,
    Searchstring  = &text,
    Filtype         = docx,
    debug           = Y);
```



Search Result for Any Files Containing: mockup
Search in Root Folder: C:\Oncology\PharmaSUG2019\test\testcase4
(Ran at 2019-02-18T19:41:00)

File Location=C:\Oncology\PharmaSUG2019\test\testcase4

Obs	File	File Location	Sub Folder
1	ISS_Mockup_Mono_KN999_New.docx	C:\Oncology\PharmaSUG2019\test\testcase4	N

File Location=C:\Oncology\PharmaSUG2019\test\testcase4\backup

Obs	File	File Location	Sub Folder
2	ISS_Mockup_Mono_KN999_20180917.docx	C:\Oncology\PharmaSUG2019\test\testcase4\backup	Y

Figure 6. Sample output of searching specific files in a folder including subfolders.

CONCLUSION

This paper presents four useful utilities that can be used as a customized search engine to find the key information we are looking for. These utilities enable us to respond to queries in different categories in a much more efficient and effective way. We have been using these utilities to provide fast and accurate responses to numerous information requests, including those from internal stakeholders, as well as those from world-wide health agencies and investigators.

ACKNOWLEDGMENTS

The authors would like to thank Mary Varughese and Cynthia He for their great support and valuable input into this paper.

CONTACT INFORMATION

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

Name: Jeff Xia
Enterprise: Merck Address: 126 E. Lincoln Avenue
City, State ZIP: Rahway, NJ 07065-4607
Work Phone: 732-594-6439
E-mail: jeff.xia@merck.com
Web: www.merck.com

Name: Shunbing Zhao
Enterprise: Merck Address: 126 E. Lincoln Avenue
City, State ZIP: Rahway, NJ 07065-4607
Work Phone: 732-594-3976
E-mail: shunbing.zhao@merck.com
Web: www.merck.com

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.