PharmaSUG 2020 - Paper AD-347

A Tool to Organize SAS Programs, Output and More for a Clinical Study

Yang Gao, Merck & Co., Inc., Rahway, NJ, USA

ABSTRACT

Depending on the complexity of a study, there can be many programs and outputs to manage and review or even rerun due to new data or requirements. Programmers must navigate to different program subfolders to select and run the affected programs and revalidate as required.

This paper presents a utility macro which can substantially reduce the resource and time spent on this task. This utility macro captures all SAS programs or output from a specified folder and saves them in an Excel Spreadsheet with the information of file path and corresponding hyperlink, which provides a convenient way to navigate to individual folders and subfolders. The hyperlinks facilitate output review for programmers and statisticians when there is a need for updates. In addition, this macro utility includes functionality to list the output with the corresponding SAS programs in the same spreadsheet. Programmers can quickly find the corresponding programs and outputs. Furthermore, this utility can be used to generate a script for batch run eliminating the need of hardcoding the program names manually. The title of output tables provides additional information for quick review. This utility macro improves programming efficiency by reducing manual effort and eliminating potential errors during the SAS program updating process. This utility is also useful for reviewing work completed by a partner and assisting managers to estimate workload.

INTRODUCTION

Managing and reviewing all SAS programs and outputs for a database lock is challenging, and often overwhelming, since additional programs continue to be added as study data is prepared for the filing the final clinical study report (CSR). Traditionally, a labor-intensive manual process that requires programmers to navigate to a folder or subfolder, check or update each file, and then review code and output is used to ensure all information is complete and accurate. This manual process can result in mistakes including: an incorrect total number of outputs being produced, and the review of some output being overlooked. This manual process is not an effective or efficient way to ensure high quality programs and outputs. It usually requires more resources and time to find, run, update, and review all the information. Therefore, a utility macro that could facilitate this time-consuming process would be very desirable.

This paper introduces a macro, which can retrieve all required information for a study and store it in an Excel spreadsheet with embedded hyperlinks. This utility macro has many functionalities:

- Facilitates the review process by listing all locations for programs and corresponding tables, figures and listings
- Creates a batch run program in a logical running order for a database lock
- Links programs to corresponding outputs when they reside in one location after a batch run
- Generates an Excel spreadsheet
- Provides hyperlinks for files and associated folders, which significantly reduces the time to navigate to different folders
- Provides the title for each table, figure and listing to facilitate review
- Serves as a cross-reference for linking all macros, logs, and outputs at different programming stages

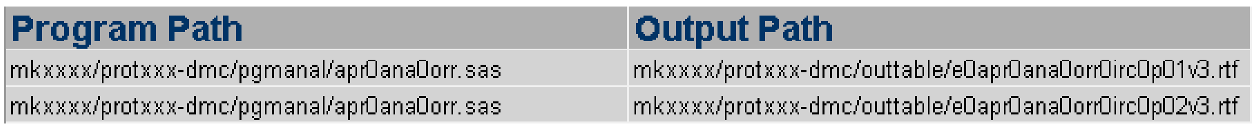
- Estimate the workload for the study programming team, to identify gaps between the work of the study team and the standard team, and to plan upcoming studies.

STEP 1: RETRIEVE ALL SAS PROGRAMS OR OUTPUT FILES WITHIN A DIRECTORY INCLUDING SUBDIRECTORIES.

With the traditional process, if programming specifications are updated, a programmer must typically go to a series of folders and subfolders to find the specific program prior to updating it. Programmers then rerun and check the log files and outputs by navigating to other subfolders. When the updated specification affects all programs, the time spent on this task is significantly increased. To improve the efficiency of this process and eliminate manual errors, using a SAS utility macro to automatically retrieve all programs for a protocol is an attractive option.

This macro has the functionality to capture all files with different extensions, such as .sas, .rtf, .sas7bdat, .log, .lst and .xls within the directory including any subdirectories. There are two parameters associated with this macro: &dir is the full path of the directory and &ext is the file extension. If no extension is specified, it assumes that the name is a directory. The embedded call program is used to ensure that all subdirectories are read and all files with the specified extension are retrieved. The key part of the SAS code is shown below:

```
%macro filedirec(dir=, ext=);
%let rc=%sysfunc(filename(filrf, &dir));
%let did=%sysfunc(dopen(&filrf));

%do i = 1 %to %sysfunc(dnum(&did));
   %let name=%qsysfunc(dread(&did, &i));

   %if %qupcase(%qscan(&name, -1, .)) = %upcase(&ext) %then
         %do;

               proc sql;
                     insert into subfold
                     values("&dir/&name");
               quit;

         %end;

   %if %qscan(&name, 2, .) = %then
         %do;
               %filedirec(dir=&dir/%unquote(&name), ext=&ext)
         %end;
%end;
%mend filedirec;
```

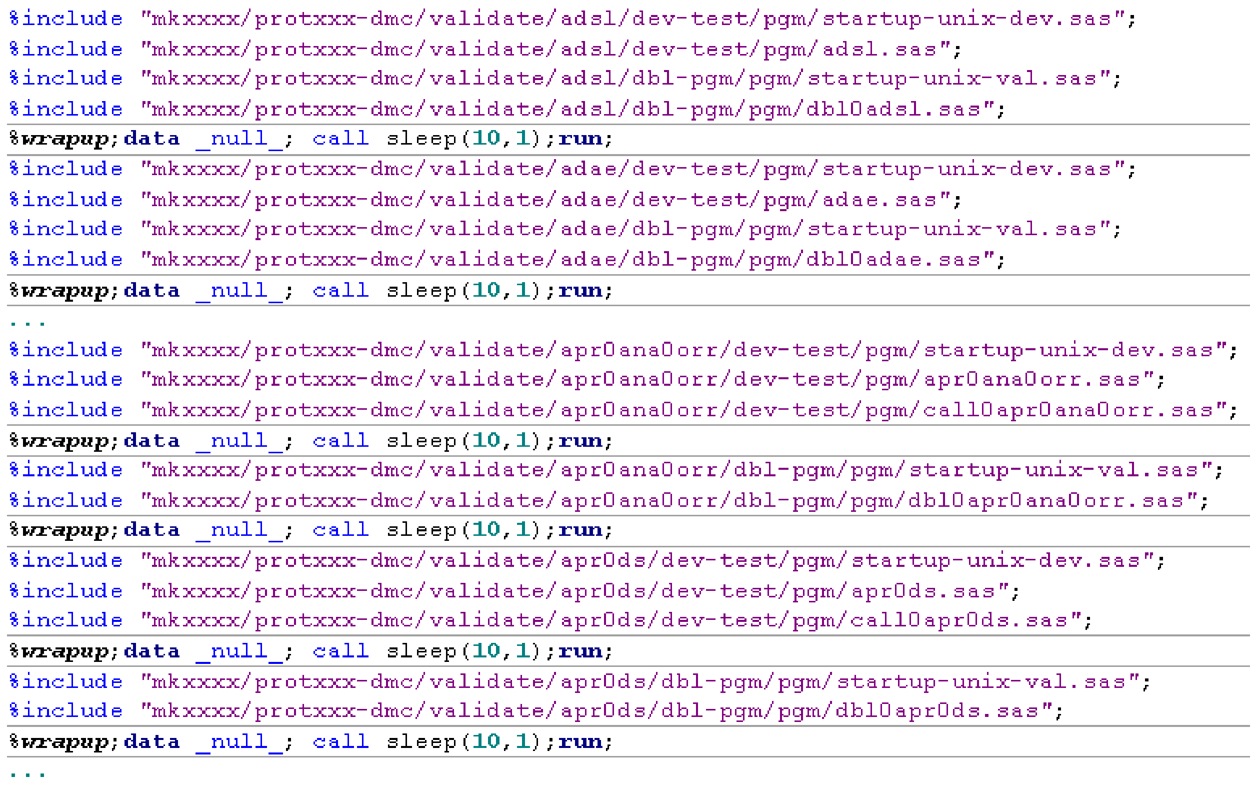


Program Path	Output Path
mkxxxx/protxxx-dmc/pgmanal/apr0ana0orr.sas	mkxxxx/protxxx-dmc/outtable/e0apr0ana0orr0irc0p01v3.rtf
mkxxxx/protxxx-dmc/pgmanal/apr0ana0orr.sas	mkxxxx/protxxx-dmc/outtable/e0apr0ana0orr0irc0p02v3.rtf

Figure 1. An Example of Retrieved Program and Outputs

STEP 2: CREATE A SAS FILE TO RUN ALL PROGRAMS IN BATCH MODE.

Once the extracted SAS programs in a directory are saved in a dataset, the next step for this tool is to create a batch file that includes all necessary SAS programs in a logical running order. Creation of a batch file enables a one-time submission of the batch program. Very often the lead programmer needs to create a run-all file by typing in all the program names. For studies with many programs, the manual creation of a batch file can take a tremendous amount of time and increase the possibility of errors. This easy-to-use utility macro can create a batch file with a logical running order. It is not necessary for programmers to consider or adjust the running dependencies when performing a batch run. Figure 2 is an example of a batch file.



```
%include "mkxxxx/protxxx-dmc/validate/adsl/dev-test/pgm/startup-unix-dev.sas";
%include "mkxxxx/protxxx-dmc/validate/adsl/dev-test/pgm/adsl.sas";
%include "mkxxxx/protxxx-dmc/validate/adsl/dbl-pgm/pgm/startup-unix-val.sas";
%include "mkxxxx/protxxx-dmc/validate/adsl/dbl-pgm/pgm/dbl0adsl.sas";
%wrapup;data _null_; call sleep(10,1);run;
%include "mkxxxx/protxxx-dmc/validate/adae/dev-test/pgm/startup-unix-dev.sas";
%include "mkxxxx/protxxx-dmc/validate/adae/dev-test/pgm/adae.sas";
%include "mkxxxx/protxxx-dmc/validate/adae/dbl-pgm/pgm/startup-unix-val.sas";
%include "mkxxxx/protxxx-dmc/validate/adae/dbl-pgm/pgm/dbl0adae.sas";
%wrapup;data _null_; call sleep(10,1);run;
...
%include "mkxxxx/protxxx-dmc/validate/apr0ana0orr/dev-test/pgm/startup-unix-dev.sas";
%include "mkxxxx/protxxx-dmc/validate/apr0ana0orr/dev-test/pgm/apr0ana0orr.sas";
%include "mkxxxx/protxxx-dmc/validate/apr0ana0orr/dev-test/pgm/call0apr0ana0orr.sas";
%wrapup;data _null_; call sleep(10,1);run;
%include "mkxxxx/protxxx-dmc/validate/apr0ana0orr/dbl-pgm/pgm/startup-unix-val.sas";
%include "mkxxxx/protxxx-dmc/validate/apr0ana0orr/dbl-pgm/pgm/dbl0apr0ana0orr.sas";
%wrapup;data _null_; call sleep(10,1);run;
%include "mkxxxx/protxxx-dmc/validate/apr0ds/dev-test/pgm/startup-unix-dev.sas";
%include "mkxxxx/protxxx-dmc/validate/apr0ds/dev-test/pgm/apr0ds.sas";
%include "mkxxxx/protxxx-dmc/validate/apr0ds/dev-test/pgm/call0apr0ds.sas";
%wrapup;data _null_; call sleep(10,1);run;
%include "mkxxxx/protxxx-dmc/validate/apr0ds/dbl-pgm/pgm/startup-unix-val.sas";
%include "mkxxxx/protxxx-dmc/validate/apr0ds/dbl-pgm/pgm/dbl0apr0ds.sas";
%wrapup;data _null_; call sleep(10,1);run;
...
```

Figure 2. The Batch Run Program with A Logical Running Order

STEP 3: OUTPUT THE RETRIEVED DIRECTORIES TO AN XML BASED EXCEL FILE.

The SAS dataset containing full paths can be output to an Excel file for documentation and future reference. The SAS ODS ExcelXP Tagset is widely utilized to generate XML output, which is compatible with Microsoft Excel. The syntax is shown below. Path is the full path where the XML file is stored. File specifies the external file that contains the XML output. The SAS code is shown below:

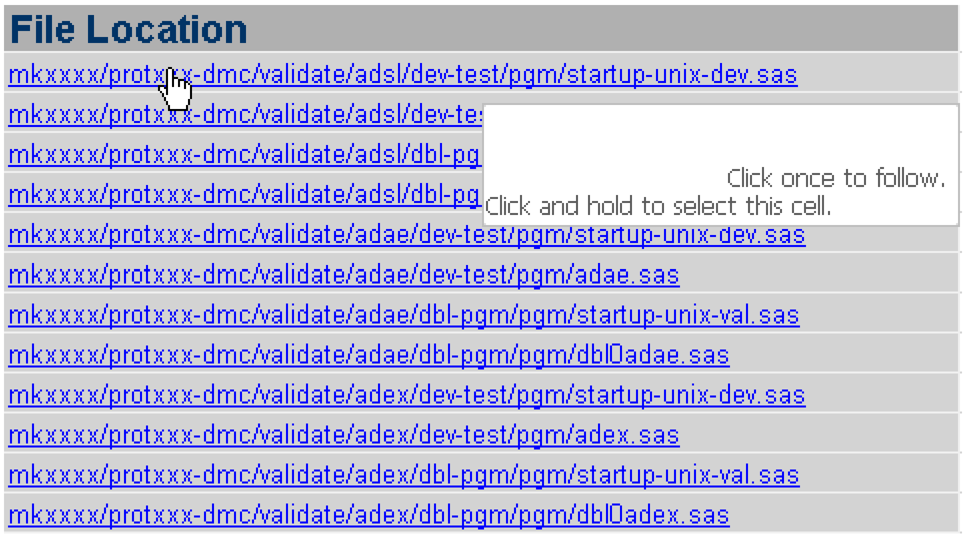
```
ods tagsets.excelxp path="the-desired-directory" (url=none)
   file="protocolxxx-dmc.xml";
   proc report data=forlink;
      ...
   run;
ods tagsets.excelxp close;
```

STEP 4: ESTABLISH HYPERLINKS FOR THE FILE LOCATION.

Hyperlinks provide direct access to the SAS program location or the output destination, and it allows programmers to directly navigate to the point of interest with a simple click. The hyperlinks could route to a folder or a file. The Excel file with clickable hyperlinks serves as a central location that contains all SAS programs and output files for a study.

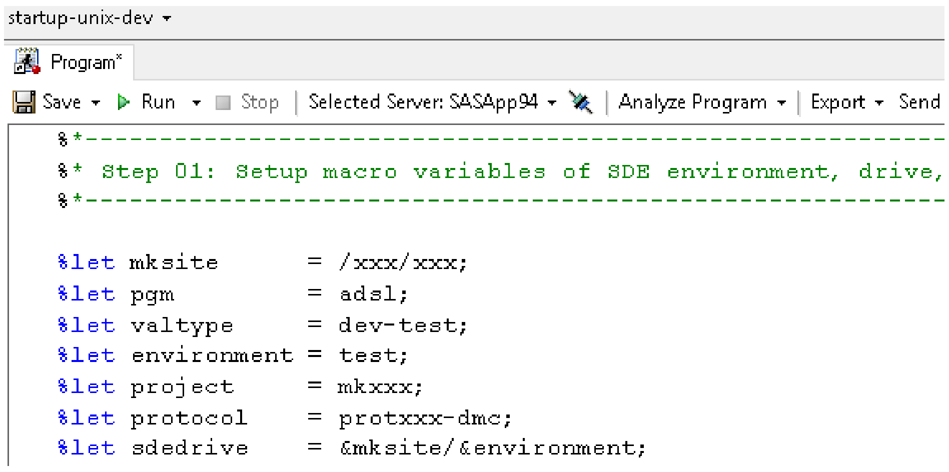
There are a variety of ways to use the SAS system to build hyperlinks. This utility macro uses the COMPUTE block and CALL DEFINE routine in PROC REPORT to create dynamic links. There are two CALL DEFINE statements in one COMPUTE block. One requires a valid URL, while the other one is used to specify the STYLE of the displayed hyperlink. Sample code and the resulting Excel sheet are shown in Figure 3 below:

```
  proc report data=forlink nowd;
       column filepath;
       define filepath  / "File Location" style(header)=[just=l]
                          style(column)=[just=left asis=on];

       compute filepath;
          call define(_col_,'URL',filepath);
          call define(_col_,'style','style={textdecoration=underline
color=blue}');
       endcomp;
   run;
```



File Location
mkxxxx/protxxx-dmc/validate/adsl/dev-test/pgm/startup-unix-dev.sas
mkxxxx/protxxx-dmc/validate/adsl/dev-tes
mkxxxx/protxxx-dmc/validate/adsl/dbl-pg
mkxxxx/protxxx-dmc/validate/adsl/dbl-pg
mkxxxx/protxxx-dmc/validate/adae/dev-test/pgm/startup-unix-dev.sas
mkxxxx/protxxx-dmc/validate/adae/dev-test/pgm/adae.sas
mkxxxx/protxxx-dmc/validate/adae/dbl-pgm/pgm/startup-unix-val.sas
mkxxxx/protxxx-dmc/validate/adae/dbl-pgm/pgm/dbl0adae.sas
mkxxxx/protxxx-dmc/validate/adex/dev-test/pgm/startup-unix-dev.sas
mkxxxx/protxxx-dmc/validate/adex/dev-test/pgm/adex.sas
mkxxxx/protxxx-dmc/validate/adex/dbl-pgm/pgm/startup-unix-val.sas
mkxxxx/protxxx-dmc/validate/adex/dbl-pgm/pgm/dbl0adex.sas

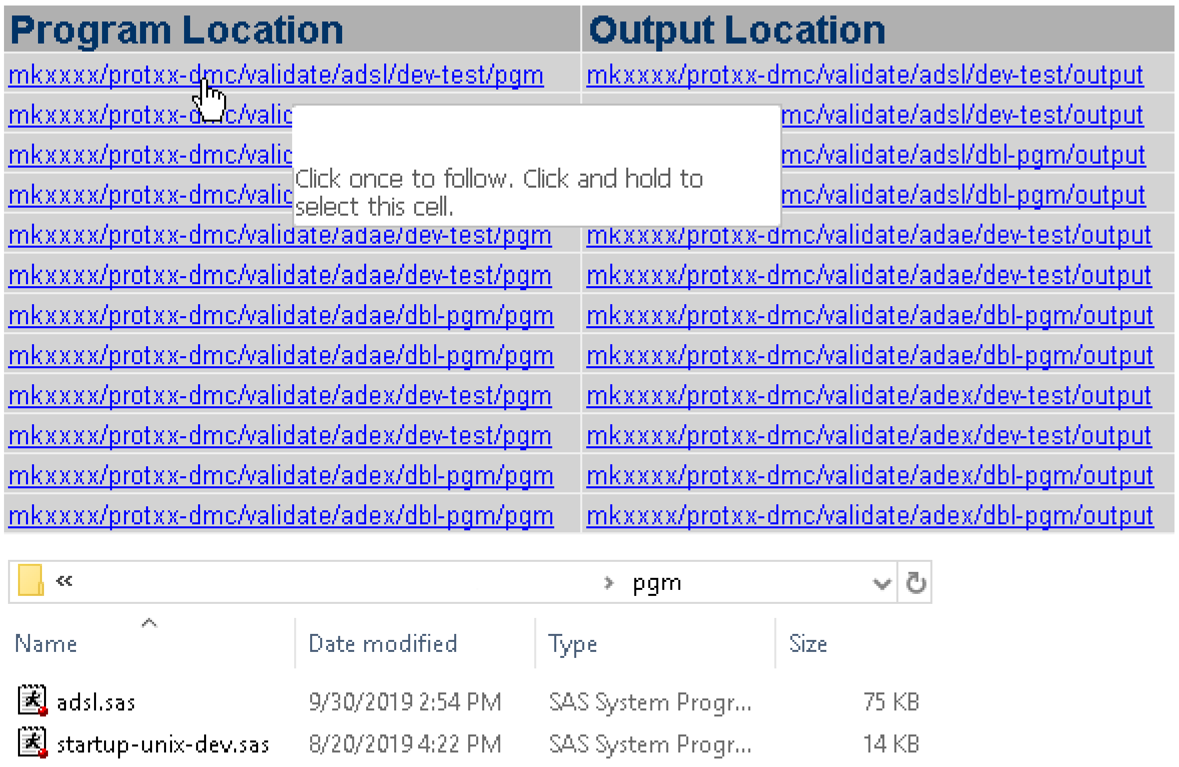
Click once to follow.
Click and hold to select this cell.


startup-unix-dev
Program*
Save Run Stop Selected Server: SASApp94 Analyze Program Export Send

```
%*------------------------------------------------------------
%* Step 01: Setup macro variables of SDE environment, drive,
%*------------------------------------------------------------

%let mksite      = /xxx/xxx;
%let pgm         = adsl;
%let valtype     = dev-test;
%let environment = test;
%let project     = mkxxx;
%let protocol    = protxxx-dmc;
%let sdedrive    = &mksite/&environment;
```

Figure 3. Hyperlinks of SAS Programs

In addition to the links to specific SAS programs, the Excel file contains hyperlinks to the folders that contain the SAS programs. This facilitates navigation to those folders in the event that date and time stamps of files need to be reviewed. The links navigating to the output file location are useful for output review and log check. The hyperlinks in the Excel file significantly improve the efficiency and quality of a programmer's routine work.



Program Location	Output Location
mkxxxx/protxx-dmc/validate/adsl/dev-test/pgm	mkxxxx/protxx-dmc/validate/adsl/dev-test/output
mkxxxx/protxx-d c/valic	mc/validate/adsl/dev-test/output
mkxxxx/protxx-dmc/valic	mc/validate/adsl/dbl-pgm/output
mkxxxx/protxx-dmc/valic	mc/validate/adsl/dbl-pgm/output
mkxxxx/protxx-dmc/validate/adae/dev-test/pgm	mkxxxx/protxx-dmc/validate/adae/dev-test/output
mkxxxx/protxx-dmc/validate/adae/dev-test/pgm	mkxxxx/protxx-dmc/validate/adae/dev-test/output
mkxxxx/protxx-dmc/validate/adae/dbl-pgm/pgm	mkxxxx/protxx-dmc/validate/adae/dbl-pgm/output
mkxxxx/protxx-dmc/validate/adae/dbl-pgm/pgm	mkxxxx/protxx-dmc/validate/adae/dbl-pgm/output
mkxxxx/protxx-dmc/validate/adex/dev-test/pgm	mkxxxx/protxx-dmc/validate/adex/dev-test/output
mkxxxx/protxx-dmc/validate/adex/dev-test/pgm	mkxxxx/protxx-dmc/validate/adex/dev-test/output
mkxxxx/protxx-dmc/validate/adex/dbl-pgm/pgm	mkxxxx/protxx-dmc/validate/adex/dbl-pgm/output
mkxxxx/protxx-dmc/validate/adex/dbl-pgm/pgm	mkxxxx/protxx-dmc/validate/adex/dbl-pgm/output

Click once to follow. Click and hold to select this cell.

« › pgm

Name	Date modified	Type	Size
adsl.sas	9/30/2019 2:54 PM	SAS System Progr...	75 KB
startup-unix-dev.sas	8/20/2019 4:22 PM	SAS System Progr...	14 KB

Figure 4. Hyperlinks of SAS Program Folder and Output Folder Generated in Excel

STEP 5: TIE PROGRAMS AND CORRESPONDING OUTPUTS.

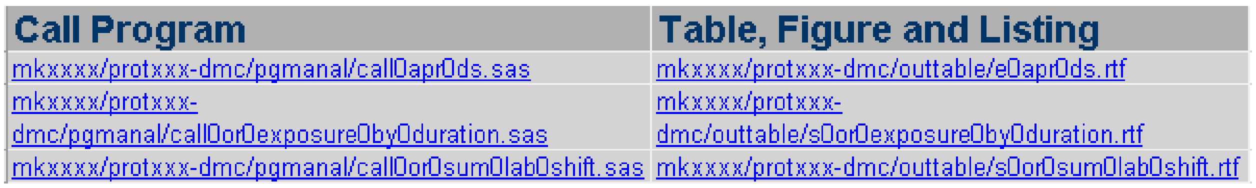
Programmers are often asked which macro generated which output after a batch run of programs since all outputs are put into the same folder. For example, very often statisticians request the call program that generated a specified output during the review process. If there is nothing to cross reference the call program to the output, traceability may be compromised.

There are a few options to resolve this issue. One option is to use a manual programming tracker. The drawbacks of using a manual programming tracker include typographical errors and out-of-date information. The reliability and accuracy of the programming tracker are undermined by its manual creation. However, using a utility macro that programmatically scans through the call program body and extracts output names by searching specific text strings can alleviate these concerns. Programmers and statisticians do not need to manually check files and relay this information to each other in order to identify call programs and corresponding outputs after running this utility macro. This macro cross-references programs and outputs and provides the necessary traceability of workflow for programmers. This feature is especially useful for new hires and for those taking over the work from others because everything is clearly linked by the macro.

SCENARIO 1: CALL PROGRAM AND OUTPUT NAMES ARE CONSISTENT.

This macro first retrieves all file names from program and output folders using the same macro in STEP 1. It then scans through the call programs using a DO WHILE statement to repetitively search the text strings until the end of the program. The desired lines are stored in a dataset, which holds the output names extracted from the call program. The STEP 1 macro is called an additional time to create another dataset that contains the output names from the output repository. After that, these two datasets can be merged or joined based on their common output names. The programs and corresponding outputs are tied, and the traceability has been built.

```
data textfromcall;
   length filepath _filepath $200.;
   infile sfile filename = _filepath lrecl=200 end = done;
      do while (not done);
      input rec $200.;
      filepath = strip(_filepath);
      output;
      end;
run;
```

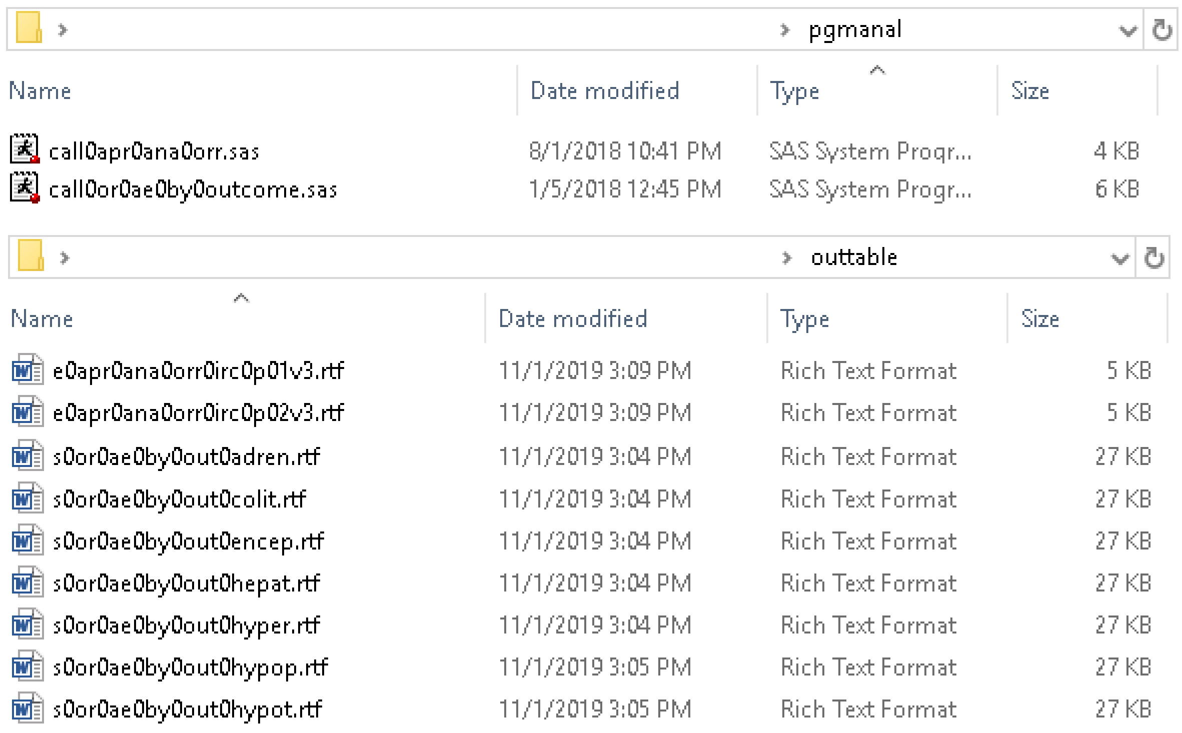


Call Program	Table, Figure and Listing
mkxxxx/protxxx-dmc/pgmanal/call0apr0ds.sas	mkxxxx/protxxx-dmc/outtable/e0apr0ds.rtf
mkxxxx/protxxx-dmc/pgmanal/call0or0exposure0by0duration.sas	mkxxxx/protxxx-dmc/outtable/s0or0exposure0by0duration.rtf
mkxxxx/protxxx-dmc/pgmanal/call0or0sum0lab0shift.sas	mkxxxx/protxxx-dmc/outtable/s0or0sum0lab0shift.rtf

Figure 5. Link Call Programs and Output When Names Match.

SCENARIO 2: CALL PROGRAM AND OUTPUT NAMES ARE NOT CONSISTENT.

In many cases, the call program and output files names are not identical which makes joining by name impossible. When the names of call programs and corresponding output do not match, it’s difficult for programmers to identify them quickly and accurately. There are often valid reasons that names do not match, such as the parameter values of output name in call programs are passed with macro variables, or there are subsets for outputs. The following pictures show some examples. The outputs are created from the following two call programs, but they cannot be merged together because there is no common value for the file names.



pgmanal

Name	Date modified	Type	Size
call0apr0ana0orr.sas	8/1/2018 10:41 PM	SAS System Progr...	4 KB
call0or0ae0by0outcome.sas	1/5/2018 12:45 PM	SAS System Progr...	6 KB

outtable

Name	Date modified	Type	Size
e0apr0ana0orr0irc0p01v3.rtf	11/1/2019 3:09 PM	Rich Text Format	5 KB
e0apr0ana0orr0irc0p02v3.rtf	11/1/2019 3:09 PM	Rich Text Format	5 KB
s0or0ae0by0out0adren.rtf	11/1/2019 3:04 PM	Rich Text Format	27 KB
s0or0ae0by0out0colit.rtf	11/1/2019 3:04 PM	Rich Text Format	27 KB
s0or0ae0by0out0encep.rtf	11/1/2019 3:04 PM	Rich Text Format	27 KB
s0or0ae0by0out0hepat.rtf	11/1/2019 3:04 PM	Rich Text Format	27 KB
s0or0ae0by0out0hyper.rtf	11/1/2019 3:04 PM	Rich Text Format	27 KB
s0or0ae0by0out0hypop.rtf	11/1/2019 3:05 PM	Rich Text Format	27 KB
s0or0ae0by0out0hypot.rtf	11/1/2019 3:05 PM	Rich Text Format	27 KB

Figure 6. Examples Where Call Program and Output Do Not Match.

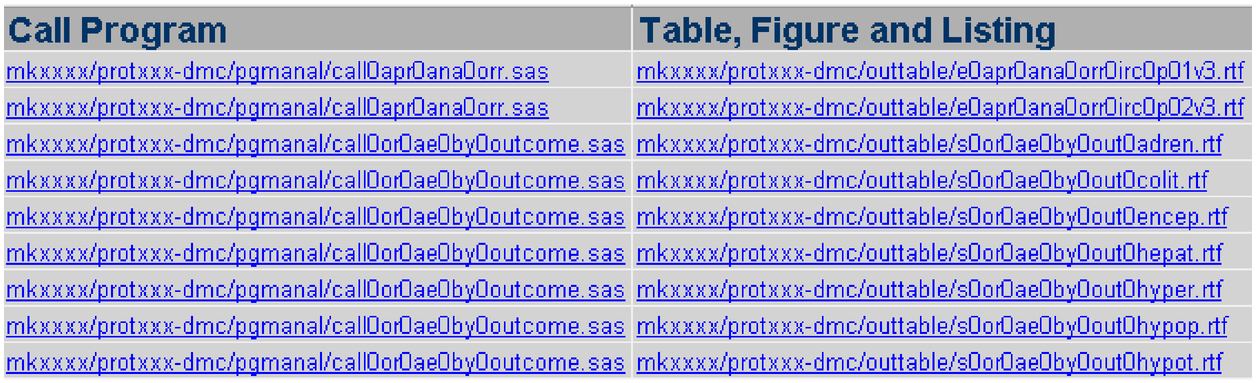
It is impossible to tie two files using MERGE or JOIN if the names are not identical. If we want to associate the call program and corresponding outputs, we need to make matching character strings more flexible. Within SAS, there are several functions that allow programmers to perform a fuzzy match. This utility macro uses the COMPGED function. Specifically, the COMPGED function returns a generalization of the levenshtein edit distance, which is a measure of dissimilarity between two strings. A lower score indicates two strings are more similar than other strings that have higher scores.

To perform the fuzzy match, this macro first uses PROC SQL to create the cartesian product when joining two tables. This means it builds all possible combinations between the two tables based on names. After that, calculated values by using COMPGED function are ordered within a group. The call program having the minimum value is selected as the one that creates the specific output. The following SAS code can handle these cases.

```
proc sql noprint;
   create table newdata1 as
   select a. foldpath1, a. outname1, b. foldpath2, b. outname2, compged
          (outname1, outname2) as cost
   from output a, pgmanal b
   order by outname1, outname2;
quit;

proc sql noprint;

   create table newdata2 as
```

```
      select a.*, min (cost) as costmin
      from newdata1 a
      group by outname1;
   quit;
```

After using the fuzzy match, the outputs in Figure 6 have been associated to the closest matching call programs. The accuracy of results depends on the similarity of the two strings from two sources. The hyperlinks of the call programs and corresponding output are created following STEP 4.

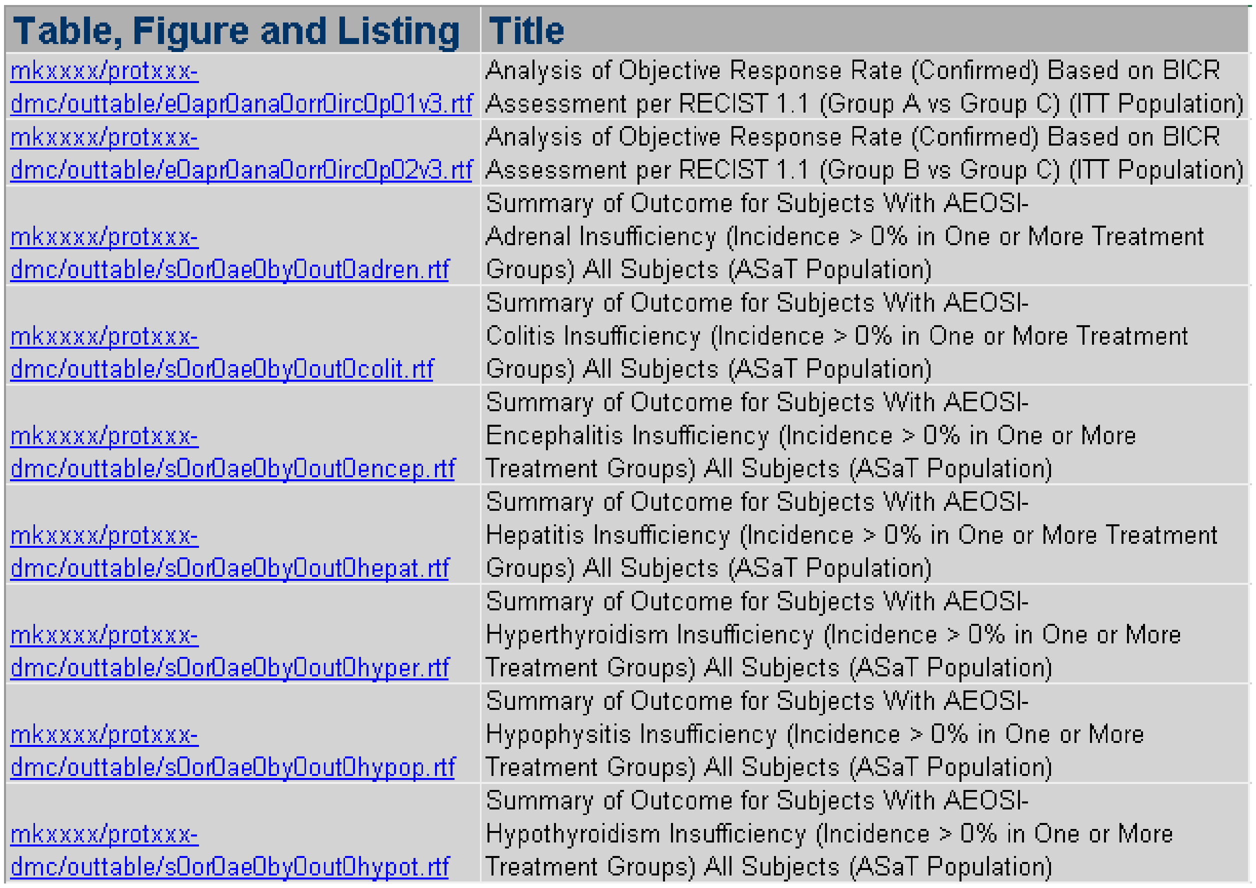


Call Program	Table, Figure and Listing
mkxxxx/protxxx-dmc/pgmanal/call0apr0ana0orr.sas	mkxxxx/protxxx-dmc/outtable/e0apr0ana0orr0irc0p01v3.rtf
mkxxxx/protxxx-dmc/pgmanal/call0apr0ana0orr.sas	mkxxxx/protxxx-dmc/outtable/e0apr0ana0orr0irc0p02v3.rtf
mkxxxx/protxxx-dmc/pgmanal/call0or0ae0by0outcome.sas	mkxxxx/protxxx-dmc/outtable/s0or0ae0by0out0adren.rtf
mkxxxx/protxxx-dmc/pgmanal/call0or0ae0by0outcome.sas	mkxxxx/protxxx-dmc/outtable/s0or0ae0by0out0colit.rtf
mkxxxx/protxxx-dmc/pgmanal/call0or0ae0by0outcome.sas	mkxxxx/protxxx-dmc/outtable/s0or0ae0by0out0encep.rtf
mkxxxx/protxxx-dmc/pgmanal/call0or0ae0by0outcome.sas	mkxxxx/protxxx-dmc/outtable/s0or0ae0by0out0hepat.rtf
mkxxxx/protxxx-dmc/pgmanal/call0or0ae0by0outcome.sas	mkxxxx/protxxx-dmc/outtable/s0or0ae0by0out0hyper.rtf
mkxxxx/protxxx-dmc/pgmanal/call0or0ae0by0outcome.sas	mkxxxx/protxxx-dmc/outtable/s0or0ae0by0out0hypop.rtf
mkxxxx/protxxx-dmc/pgmanal/call0or0ae0by0outcome.sas	mkxxxx/protxxx-dmc/outtable/s0or0ae0by0out0hypot.rtf

Figure 7. Examples of Call Programs and Outputs That Are Associated Using Fuzzy Match

STEP 6: EXTRACT TITLES FROM RTF.

Adding a title column on the side of a table, figure or listing provides additional details and allows statisticians and programmers to quickly review the deliverables. This can be completed by applying similar SAS code in STEP 5. The difference is that the data prior to the DATA step is in an RTF file instead of a SAS dataset. After obtaining the desired text strings from the RTF file, those strings can be concatenated to create a complete title. The following picture shows an example.



Table, Figure and Listing	Title
mkxxxx/protxxx-dmc/outtable/e0apr0ana0orr0irc0p01v3.rtf	Analysis of Objective Response Rate (Confirmed) Based on BICR Assessment per RECIST 1.1 (Group A vs Group C) (ITT Population)
mkxxxx/protxxx-dmc/outtable/e0apr0ana0orr0irc0p02v3.rtf	Analysis of Objective Response Rate (Confirmed) Based on BICR Assessment per RECIST 1.1 (Group B vs Group C) (ITT Population)
mkxxxx/protxxx-dmc/outtable/s0or0ae0by0out0adren.rtf	Summary of Outcome for Subjects With AEOSI-Adrenal Insufficiency (Incidence > 0% in One or More Treatment Groups) All Subjects (ASaT Population)
mkxxxx/protxxx-dmc/outtable/s0or0ae0by0out0colit.rtf	Summary of Outcome for Subjects With AEOSI-Colitis Insufficiency (Incidence > 0% in One or More Treatment Groups) All Subjects (ASaT Population)
mkxxxx/protxxx-dmc/outtable/s0or0ae0by0out0encep.rtf	Summary of Outcome for Subjects With AEOSI-Encephalitis Insufficiency (Incidence > 0% in One or More Treatment Groups) All Subjects (ASaT Population)
mkxxxx/protxxx-dmc/outtable/s0or0ae0by0out0hepat.rtf	Summary of Outcome for Subjects With AEOSI-Hepatitis Insufficiency (Incidence > 0% in One or More Treatment Groups) All Subjects (ASaT Population)
mkxxxx/protxxx-dmc/outtable/s0or0ae0by0out0hyper.rtf	Summary of Outcome for Subjects With AEOSI-Hyperthyroidism Insufficiency (Incidence > 0% in One or More Treatment Groups) All Subjects (ASaT Population)
mkxxxx/protxxx-dmc/outtable/s0or0ae0by0out0hypop.rtf	Summary of Outcome for Subjects With AEOSI-Hypophysitis Insufficiency (Incidence > 0% in One or More Treatment Groups) All Subjects (ASaT Population)
mkxxxx/protxxx-dmc/outtable/s0or0ae0by0out0hypot.rtf	Summary of Outcome for Subjects With AEOSI-Hypothyroidism Insufficiency (Incidence > 0% in One or More Treatment Groups) All Subjects (ASaT Population)

Figure 8. Outputs with Extracted Titles

CONCLUSION

This paper presents a powerful macro which handles the challenging task of managing and organizing all programs and outputs for a clinical study. This tool not only turns the traditional manual efforts to an automatic programming process, but also provides a few critical features to facilitate programmers' daily work. This macro allows extraction of file path regardless of extensions and the creation of the batch run file with a logical running order. The embedded hyperlinks provide quick access to specific files and associated folders. The links between call programs and related outputs are built for ease of tracking. The titles associated with RTF outputs provide additional information for checking and quickly reviewing deliverables. This utility macro has substantially enhanced the accuracy, reliability and efficiency of managing programs and outputs in contrast to the manual process.

REFERENCES

Qi, LJ and Donthi, B. 2019. "Automating SAS Program Table of Contents for Your FDA Submission Package". PharmaSUG

Carpenter, A.L. 2009. "Advanced PROC REPORT: Getting Your Tables Connected Using Links - Part I Tutorial". Proceedings of SAS Global Forum.

Vanam, M. Karidi, K. and Dodlapati, S. 2010. "SAS to Excel with ExcelXP Tagset ". PharmaSUG

List All Files within a Directory Including Subdirectories. https://documentation.sas.com/?docsetId=mcrolref&docsetTarget=n0js70lrkxo6uvn1fl4a5aafnlgt.htm&docsetVersion=9.4&locale=en

Russell, K. 2015. “How to perform a fuzzy match using SAS functions”. https://blogs.sas.com/content/sgf/2015/01/27/how-to-perform-a-fuzzy-match-using-sas-functions/

ACKNOWLEDGMENTS

The author would like to thank Lisa Spring, Christine Teng, Jeff Xia, Mary Varughese and

Amy Gillespie for their encouragement and thorough review of this paper.

CONTACT INFORMATION

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

Name: Yang Gao
Enterprise: Merck Address: 126 E. Lincoln Avenue
City, State ZIP: Rahway, NJ 07065-4607
Work Phone: 732-594-8120
E-mail: yang.gao5@merck.com
Web: www.merck.com