Paper AD20

Catch the Bad Guys!!! A Utility Program to Check SAS® Log Files

Amit Baid, ICON Clinical Research, Smyrna, GA

ABSTRACT

SAS log files are generated whenever a SAS program is run. Checking the log file is an important part of the validation process and enables us to make our programs error free. Usually we look for messages such as "ERROR" or "WARNING", but in this paper I am going to discuss some other messages which we also do not want to see in our log file. The utility program presented in this paper scans through one or more SAS log files in a specific folder for all the potential error messages. It provides a summary report which will assist a programmer in fixing the code to get rid of those messages. Going through this summary report is better than going through each SAS log file and finding all the unwanted messages. This program was developed on a PC SAS environment and works with both 8.x and 9.x versions of SAS.

INTRODUCTION

Validation is an important aspect in any programming project. Checking a log file which has hundreds of lines of code is a tedious task and error messages can be easily overlooked. SCANLOG is a utility program that will scan log files and provide a detailed summary report, making the programmer's task easier.

DESCRIPTION

SCANLOG is a utility program that scans one or more SAS log files present in a specific directory folder and looks for error messages. Below is a list of messages that the SCANLOG identifies and should be avoided as per good programming practice in order to maintain a clean log:

- fatal
- ERROR
- WARNING
- uninitialized
- MERGE statement has more than one data set with repeats of BY values
- values have been converted
- Note: Missing
- Note: Invalid argument
- W.D format was too small
- has 0 observations
- variables not in
- variables have conflicting attributes
- unequal
- Division by zero detected
- Mathematical operations could not be performed
- observations with duplicate key values were deleted

The macro code can easily be modified to search for other log messages of interest or remove messages that are of no interest to a programmer.

DETAILS

The SCANLOG macro performs a log check and prepares a summary report as per the steps described on the next page.

The first step is to provide the path of the directory containing the SAS log files to be checked in the macro call. For example if the SAS log files are located in the directory “C:\Temp”, the macro call would be:

```
%scanlog(path=C:\Temp);
```

SCANLOG will then obtain the name of all SAS log files located in the directory, which will be used in the summary report. SCANLOG next checks for messages such as “ERROR”, “WARNING”, “uninitialized”, “MERGE statement has more than one data set with repeats of BY values”, “values have been converted”, “has 0 observations”, “Division by zero detected”, etc, in all the SAS log files in the directory. SCANLOG uses the INDEX and SUBSTR function to search the log files for the error messages.

Finally, SCANLOG prepares a summary report (**scanlog.html**) consisting of a listing of all the log file names with error messages and the number of occurrences of each message per log file.

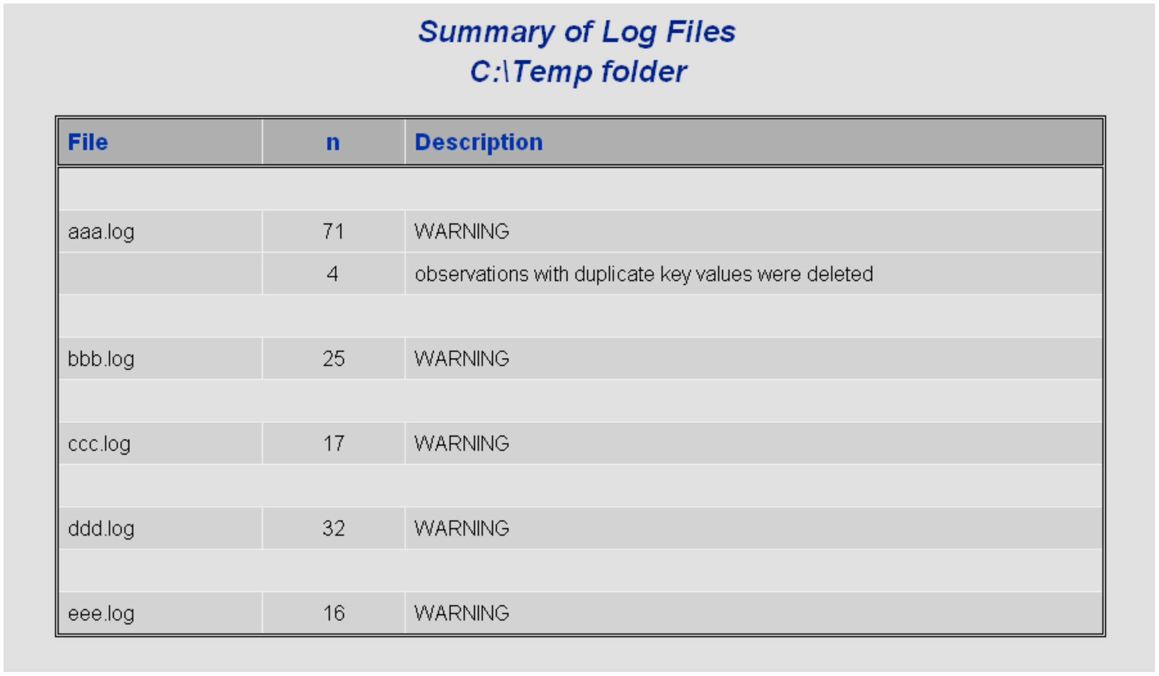
PROGRAM

```
%macro scanlog (path=);

  options nodate nonumber noxwait;

  %let path=%sysfunc(tranwrd(&path,/,\));

  ** Delete the previous version **;
  x(del &path\files.txt);

  ** Output the list of files to a temporary location **;
  x(dir /b &path\*.log >> &path\files.txt);

  ** Initialize the macro variables **;
  %let totfile=;
  %let totobs=;

  ** Get the total number of files and file names **;
  data _null_;
    infile "&path\files.txt" dsd dlm='09'x truncover;
    input name $40.;
    call symput('totfile',trim(left(put(_n_,8.))));
    call symput('file'||trim(left(put(_n_,8.))),trim(left(name)));
  run;

  %do i=1 %to &totfile;
    data files;
      infile "&path\&&file&i" dsd dlm='09'x truncover;
      input line $200.;
      length filenm $40.;
      filenm="&&file&i";

      if index(line,'fatal') then cnt1+1;
      if substr(line,1,5)='ERROR' then cnt2+1;
      if substr(line,1,7)='WARNING' then cnt3+1;
      if index(line,'uninitialized') then cnt4+1;
      if index(line,'MERGE statement has more than') then cnt5+1;
      if index(line,'values have been converted') then cnt6+1;
      if index(line,'Note: Missing') then cnt7+1;
      if index(line,'Note: Invalid argument') then cnt8+1;
      if index(line,'W.D format was too small') then cnt9+1;
      if index(line,'has 0 observations') then cnt10+1;
      if index(line,'variables not in') then cnt11+1;
      if index(line,'variables have conflicting') then cnt12+1;
      if index(line,'unequal') then cnt13+1;
      if index(line,'Division by zero detected') then cnt14+1;
      if index(line,'operations could not be performed') then cnt15+1;
      if index(line,'duplicate key values were deleted') then cnt16+1;
    run;

    proc univariate data=files noprint;
      by filenm;
      var cnt1-cnt16;
      output out=stat1 max=max1-max16;
    run;

    proc transpose data=stat1 out=stat2(where=(col1>0));
      by filenm;
      var max1-max16;
    run;

    data final;
      set %if &i>1 %then %do; final %end; stat2;
      %if &i=&totfile %then %do;
        length desc $100.;

        if lowcase(_name_)='max1' then desc='fatal';
        if lowcase(_name_)='max2' then desc='ERROR';
        if lowcase(_name_)='max3' then desc='WARNING:';
        if lowcase(_name_)='max4' then desc='uninitialized';
        if lowcase(_name_)='max5' then
          desc='Merge statement has more than one data set with repeats
                of BY values';
        if lowcase(_name_)='max6' then
          desc='values have been converted';
        if lowcase(_name_)='max7' then desc='Note: Missing';
        if lowcase(_name_)='max8' then desc='Note: Invalid argument';
        if lowcase(_name_)='max9' then desc='W.D format was too small';
        if lowcase(_name_)='max10' then desc='has 0 observations';
        if lowcase(_name_)='max11' then desc='variables not in';
        if lowcase(_name_)='max12' then
          desc='variables have conflicting attributes';
        if lowcase(_name_)='max13' then desc='unequal';
        if lowcase(_name_)='max14' then
          desc='Division by zero detected';
        if lowcase(_name_)='max15' then
          desc='Mathematical operation could not be performed';
        if lowcase(_name_)='max16' then
          desc='observations with duplicate key values were deleted';

        drop _name_ _label_;
      %end;
    run;
  %end;

  ** Get the total number of occurrences of unwanted messages **;
  data _null_;
    set final nobs=last;
    if last then call symput('totobs',trim(left(put(_n_,8.))));
  run;

  ** Final output **;
  ods listing close;
  ods noresults;
  ods html file="&path\scanlog.html";

  title1 "Summary of Log Files";
  title2 "&path folder";
```

```
  %if &totobs=0 %then %do;
    data _null_;
      file print;
      put "============================";
      put "*** SCANLOG SUCCESSFUL   ***";
      put "*** NO BAD GUYS!!! FOUND ***";
      put "============================";
    run;
  %end;
  %else %do;
    proc report data=final headline headskip split='|' missing nowd;
      columns filenm col1 desc;

      define filenm / order order=data
                      style(column)=[cellwidth=1.5in]
                      style(header)=[just=left] 'File';
      define col1   / style(column)=[cellwidth=1in]
                      center 'n' format=3.;
      define desc   / style(column)=[cellwidth=5.5in]
                      style(header)=[just=left] 'Description';
      compute before filenm;
        line ' ';
      endcomp;
    run;
  %end;

%mend scanlog;

%scanlog(path=C:\Temp);
```

OUTPUT

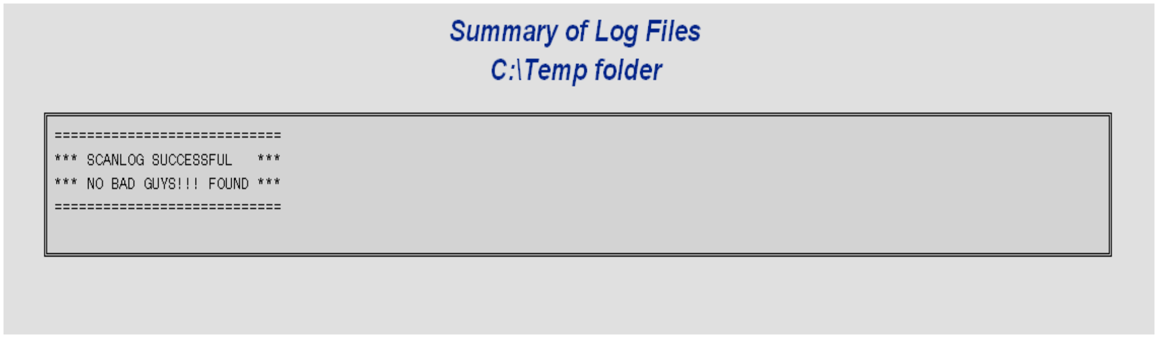
If there are any occurrences of unwanted messages then the output will be as follows:



Summary of Log Files
C:\Temp folder

File	n	Description
aaa.log	71	WARNING
	4	observations with duplicate key values were deleted
bbb.log	25	WARNING
ccc.log	17	WARNING
ddd.log	32	WARNING
eee.log	16	WARNING

If there are no occurrences of unwanted messages then the output will be as follows:



```
============================
*** SCANLOG SUCCESSFUL   ***
*** NO BAD GUYS!!! FOUND ***
============================
```

CONCLUSION

The SCANLOG macro is a useful tool that checks for various error messages in multiple log files. Using SCANLOG saves time and effort because it is no longer necessary to read through all log files line by line, thus reducing the potential of overlooking error messages that need to be resolved. SCANLOG can be easily modified to search for different error messages of interest.

REFERENCES

Carey G. Smoak (2002). Paper 96-27: A Utility Program for Checking SAS® Log Files. Proceedings of the 2002 Pharmaceutical Industry SAS Users Group.

ACKNOWLEDGEMENTS

I would like to thank Syamala Ponnapalli of ICON Clinical Research and Aimee Wahle of Sucampo Pharmaceuticals for consistently encouraging and supporting conference participation. I would also like to thank my other colleagues at ICON, particularly Vindya Somanna for providing review comments and last but not the least, my wife Sunitha for all her support.

CONTACT INFORMATION

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

Amit Baid, M.S.
ICON Clinical Research
Smyrna, GA
Phone: (619) 846-8842
Email: amit.baid@iconplc.com