Paper S-01-2012

Check and Summarize SAṢLog Files

Qiling Shi, Pittsburgh, PA

ABSTRACT

Data validation is a very important task, while validating the SAS programs is equally important to a Quality Assurance (QA) analyst. The exact positions of message such as ERRORs, WARNINGs and specific NOTEs in SAS LOG will give QA analysts a quick access to the points of possible mistakes. Checking for errors in a large SAS log file can be tedious and also easy for QA analysts to miss a lot of useful information. This paper contains a simple but very effective SAS program which can be used to check and summarize SAS log files quickly.

INTRODUCTION

There are a lot of papers containing suggestions and codes on how to check and summarize SAS log files. Checking SAS log files is very important since we need to validate the SAS programming codes and results. Usually QA analysts need to mandatorily review SAS log files. When the SAS log file has hundreds and even thousands of lines of SAS code, it is very hard for QA analysts to catch every error messages in a short time. So there are a lot of SAS program designed to reduce the burden of QA analysts and also improve the work efficiency. The code in this paper is designed to do this work also. It is simple but very effective in practice.

DESCRIPTION OF PROGRAM

This program first generates a SAS log file as an example to test the efficiency of this code. Then it needs you to input the path and name for this SAS log file. After that it uses one simple DATA STEP to check and summarize the SAS log file. Inside this DATA STEP, it has a search macro. Using this macro you can check for messages such as ERRORs, WARNINGs and specific NOTEs in the SAS LOG file. And even you can modify the code simply by using this macro to search for any string you want to find out. In this DATA STEP, it uses a PROC SUMMARY procedure to summarize the log file. PROC SUMMARY will give you the basic statistics about the messages on ERRORs, WARNINGs and specific NOTEs in the SAS LOG file.

PROGRAM CODE

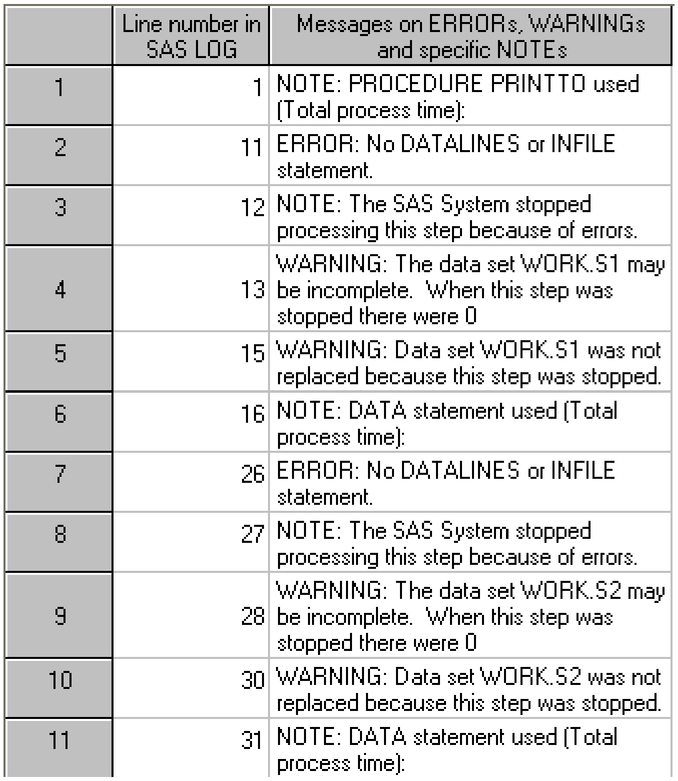
```
*create the log file as an example;
proc printto log="C:\Documents and Settings\shiq\Desktop\New
Folder\PNWSUG\test1.log" new;
run;

data s1;
  input x;
run;

data s2;
  input y;
run;

proc printto;
run;

*check and summarize the log file;
dm 'clear log; clear output;';
```

```
filename _all_ clear;

*input your log file path and names here;
filename _log_ "C:\Documents and Settings\shiq\Desktop\New
Folder\PNWSUG\test1.log";

data log1;
   label editor_line_no ='Line number in SAS LOG'
         l_r='Messages on ERRORs, WARNINGs and specific NOTEs';

   drop i1 fw string;
   retain editor_line_no 0;
   infile _log_;
   input;
   editor_line_no =editor_line_no +1;
   fw=scan(_infile_, 1, '20'x);
   if substr(fw, length(fw))=':' then;
                                 else goto bottom;
   %macro _search_(string=);
        string = "&string.";
        i1=index(upcase(_infile_), upcase("&string."));
        if i1 then goto write;
   %mend _search_;

   %_search_(string=warning:); *check for warning messages;
   %_search_(string=error:);   *check for error messages;
   %_search_(string=note:);    *check for note messages;

   write:;
     l_r=_infile_;
     output;

   bottom:;

   *summarize the log file;
     proc summary data=log1;
        class l_r;
        output out=log2;
     run;
run;
```

RESULTS

The demonstrated data set “LOG1” which is used to check the SAS file “TEST1.LOG” has 11 observations and 2 variables. One variable is called “EDITOR_LINE_NO” which contains the line number for this corresponding message in “TEST1.LOG”. The other variable “L_R” has all the messages on ERRORs, WARNINGs and specific NOTEs in “TEST1.LOG”.

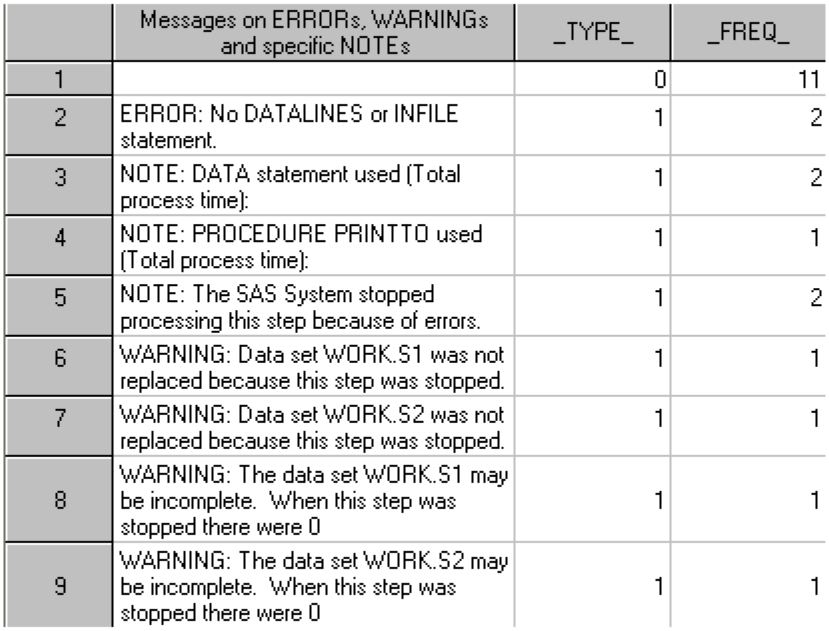
Table 1: Dataset “LOG1” checking for log file “TEST1.LOG”.



	Line number in SAS LOG	Messages on ERRORs, WARNINGs and specific NOTEs
1	1	NOTE: PROCEDURE PRINTTO used (Total process time):
2	11	ERROR: No DATALINES or INFILE statement.
3	12	NOTE: The SAS System stopped processing this step because of errors.
4	13	WARNING: The data set WORK.S1 may be incomplete. When this step was stopped there were 0
5	15	WARNING: Data set WORK.S1 was not replaced because this step was stopped.
6	16	NOTE: DATA statement used (Total process time):
7	26	ERROR: No DATALINES or INFILE statement.
8	27	NOTE: The SAS System stopped processing this step because of errors.
9	28	WARNING: The data set WORK.S2 may be incomplete. When this step was stopped there were 0
10	30	WARNING: Data set WORK.S2 was not replaced because this step was stopped.
11	31	NOTE: DATA statement used (Total process time):

The following is the summary result of table 1 which contains the basic statistic information about the messages on ERRORs, WARNINGs and specific NOTEs in “TEST1.LOG”.

Table 2: Dataset “LOG2” summarizing the checking information for log file “TEST1.LOG”.



	Messages on ERRORs, WARNINGs and specific NOTEs	_TYPE_	_FREQ_
1		0	11
2	ERROR: No DATALINES or INFILE statement.	1	2
3	NOTE: DATA statement used (Total process time):	1	2
4	NOTE: PROCEDURE PRINTTO used (Total process time):	1	1
5	NOTE: The SAS System stopped processing this step because of errors.	1	2
6	WARNING: Data set WORK.S1 was not replaced because this step was stopped.	1	1
7	WARNING: Data set WORK.S2 was not replaced because this step was stopped.	1	1
8	WARNING: The data set WORK.S1 may be incomplete. When this step was stopped there were 0	1	1
9	WARNING: The data set WORK.S2 may be incomplete. When this step was stopped there were 0	1	1

From table 2, we know that there are two error messages and several warning messages in “TEST1.LOG”. By looking into these two error messages and warning messages for more information, we can resolve the problems in SAS program more easily.

CONCLUSION

This SAS program can save time and effort for QA analysts since they don’t need to read the SAS log files line by line to find out the error messages. Also it reduces the possibility of overlooking error messages. Hence it is a very useful tool to check and summarize the SAS log files. Furthermore, it is simple, easy to understand and can be used by most QA analysts easily and efficiently.

REFERENCES

1. “Detecting Medicaid Data Anomalies Using Data Mining Techniques”, Southeast SAS Users Group Conference, 2010.
2. “Find Potential Fraud Leads Using Data Mining Techniques”, Southeast SAS Users Group Conference, 2011.
3. “Remove Voided Claims for Insurance Data”, Western Users of SAS Software Conference, 2011.
4. “Assign Overpayment to Insurance Data with Adjustments”, Southeast SAS Users Group Conference, 2011.

CONTACT INFORMATION

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

Qiling Shi, Mathematics PhD, Certified Fraud Examiner
Email: shiqiling@gmail.com
Website: http://www.linkedin.com/in/qilingshi