SESUG 2020 Paper 108

Some Korn Shell Scripts for SAS® Programmers

Hengwei Liu, Daiichi Sankyo, Inc.

ABSTRACT

A lot of pharmaceutical companies use SAS on Linux server. Linux shell scripting is a power tool for the SAS programmers. It can be used to read text files and extract information. It can be used to perform other operations on the files such as combining files or renaming files. In this paper some Korn shell scripts of interest to SAS programmers are discussed.

INTRODUCTION

Many SAS programmers run some Linux commands in their daily work, such as ls, cd, pwd and grep. For some tasks, it is worth the effort to set up a small shell script. It can be used later when such tasks arise again. A collection of shell scripts for some routine jobs improves efficiency and makes life easier for the programmers.

Both Bash and Korn shells are widely used. In this paper some Korn shell scripts are discussed. They are in two categories:

Scripts that read text files and extract information:

- A script that reads the LOG files and displays all the messages that programmers need to review.
- A script that extracts the first page of multiple LST files and combines them into a single file.
- A script that reads the LST files to identify the SAS programs and compares the timestamps of the SAS programs and the LST files.

Scripts for other file handling operations:

- A script that combines many LST files into one single file and creates a list of all the LST files.
- A script that finds SAS datasets whose names are in uppercase and renames them to lowercase.
- A script that collects the names of many SAS programs in a folder and puts them in a single file so programmers can submit all the programs at once.
- A script that converts an LST file to RTF file.

To understand the scripts in this article, the readers are expected to have basic knowledge of shell scripting, including sed, awk, pipe and command substitution.

DETAILS FOR THE SHELL SCRIPTS

Linux shell scripting is a toolbox with many tools to handle text files. The shell scripts can do many operations on the LOG files and LST files created by SAS. There is in the appendix a sample LST file to which some of the scripts in this article are applied. It is created with the SAS options PS=52, LS=132. The title with table number appears in the fourth row. The last footnote of the table shows the location of the SAS program used to create the table.

SCRIPT1.SH

In a clinical study there can be many SAS programs in a programming folder. A script can be set up to submit all of them to SAS at once. This script assumes that the SAS system is invoked by the command sas.

This script is run in the folder where the SAS programs are located. There is no command-line argument to the script. The awk command is used to get the names of all the SAS programs in a column, and sed is used to add the word sas in front of each name.

```
#!/bin/ksh
ls -l *.sas | awk '{print $9}' | sed 's/^/sas /'   > runall
chmod u+x runall
```

Output 1 shows an example of the file runall.

```
sas table1.sas
sas table2.sas
sas table3.sas
```

Output 1. Output from SCRIPT1.SH

Now the programmer can execute runall and all the SAS programs are submitted to run.

SCRIPT2.SH

The statistical tables in a clinical study are usually validated through independent programming. But a programmer may want to print out the first page of each LST file and perform a quick review. He can quickly grasp the scope of the output and identify some obvious issues before diving into all the details. This can be done through a script.

The script is run in the folder where the LST files are located. There is no command-line argument to the script. Command substitution is used to get the names of all the LST files and the head command is used to get the first page of each file.

```
#!/bin/ksh

#PS is the pagesize option used to generate the lst files
PS=52

for filename in $(ls *.lst) ; do
head -${PS} $filename >> combine
done
```

SCRIPT3.SH

In the Linux platform the SAS dataset names must be in lower case. If a programmer receives datasets whose names are in uppercase, he needs to change the names to lowercase. A script is used to change all the names from uppercase to lowercase to avoid renaming them manually.

The script is run in the folder where the SAS datasets are located. There is no command-line argument to the script. Command substitution with the tr command is used to get the new name in lower case. The mv command is used to rename the files.

```
#!/bin/ksh
for filename in $(ls *.SAS7BDAT *.sas7bdat) ; do
newfilename=$(print $filename | tr '[A-Z]' '[a-z]')
print "$filename -> $newfilename"
mv $filename $newfilename
done
```

SCRIPT4.SH

There might be hundreds of tables in a clinical study. To facilitate review, programmers are often asked to combine multiple LST files into a single file and generate a list of the LST files. This can be done in a script.

This script is run in the folder where the LST files are located. There is no command-line argument to the script. The sed command is used to extract the fourth line of the table, which is a title containing table number. Both the title and the name of each LST file are printed out in a file. This file is sorted and all the LST files are combined into a single file according to the order in the sorted file.

```
#!/bin/ksh
for filename in $(ls *.lst) ; do
print $(sed -n '4p' $filename \
| awk '{printf "%s", $0 }') $filename >> toc_temp
done

sort toc_temp > toc

for file in $(awk '{print $NF}' toc)
do
cat $file >> combine
done
```

Output 2 shows an example of the file toc.

```
Table 14.1 Demographics demog.lst
Table 14.2 Frequency of Dose Delay dose_delay.lst
Table 14.3 TEAE by Preferred Term pref.lst
Table 14.4 Time to Response by ICR resp.lst
```

Output 2. Output from SCRIPT4.SH

SCRIPT5.SH

In each LST file, the last footnote starts with the word "Program:', which is followed by the full path of the SAS program used to generate the LST file. During the study the SAS programs are often updated. Sometimes it is of concern that a program may have been updated but is not run and, as a result, the corresponding LST file is not updated.

A shell script is used to pair up the LST files and SAS programs and compare their timestamps. If the timestamp for an LST file is prior to the timestamp of the SAS program, there is an issue.

The script is run in the folder where the LST files are located. There is no command-line argument to the script. Regular expression is used to find the footnote that starts with the word "Program:". The SAS program names extracted from the footnotes are put into the file toc1. The stat command is used on each SAS program to get the timestamps and put them in the file toc2. The names of the LST files are put in the file toc3. The stat command is used again to get the timestamps of the LST files and put them in the file toc4. The four files are combined into one file with the paste command.

```
#!/bin/ksh
for filename in $(ls *.lst) ; do
awk '/^Program:/{print $0}' $filename | sed 's/Program:/ /' \
| sed 's!.*/!!' >> toc1
awk '/^Program:/{print $0}' $filename | sed 's/Program://' \
| xargs stat -c '%.19y'   >> toc2
echo $filename >> toc3
stat -c '%.19y' $filename >> toc4
done

paste -d ' ' toc1 toc2 toc3 toc4 > toc5
```

Output 3 shows an example of the file toc5.

```
delay.sas 2020-07-25 16:29:45 delay.lst 2020-07-25 16:34:42
t1401.sas 2020-07-25 16:30:03 t1401.lst 2020-07-25 16:35:01
t1402.sas 2020-07-25 16:30:05 t1402.lst 2020-07-25 16:35:12
```

Output 3. Output from SCRIPT5.SH

SCRIPT6.SH

Review of the LOG files is an important part of SAS programming work. Programmers need to review the errors, warnings and some other messages from the log files, e.g., merge statement has more than one data set with repeats of by values. When there are a lot of LOG files in a folder, it becomes necessary to use some tool to read the LOG files and collect all the messages that need to be reviewed.

Some companies use a SAS program to read the LOG files and print out the messages that require review. This task can be done through a shell script. The script reads the LOG files and print out a line if it contains certain strings.

This script is run in the folder where the LOG files are located. There is no command-line argument to the script. The egrep command is used to find all the messages that need to be reviewed. These messages are printed out under the name of the LOG file.

```
#!/bin/ksh
for saslog in $(ls *.log); do
      print $saslog
      egrep -i -n "(unclosed do|Error|Warning|values have been converted \
|statement not executed due to noexec option \
|division by zero detected at line \
|data step stopped due to looping|missing values were generated \
|uninitial|not replaced|acceptable|outside the axis range|invalid \
|not in the report definition|w.d. format|authorization|misspelled \
|is unknow|merge statement has more than one|could not be performed \
|query requires remerging|SAS System stopped processing \
|does not exist|truncated|multiple lengths were specified \
|was not found|could not be loaded)" \
$saslog
Done
```

SCRIPT7.SH

When a study team reviews the tables and listings, RTF is often the preferred format. The programmers can use ODS RTF to create the tables and listings in RTF format. A different approach is to create the output in LST format and convert them to RTF format.

SAS macro can be used to convert text files to RTF file (Franklin, 2010). This macro is adapted with minor modification into a shell script.

The shell script is run in the folder where the LST file located. It takes two arguments, the LST file name and the RTF filename. The sed command is used to replace \o14, the octal value for form feed, by \page. The awk command takes different action when there is the word \page or otherwise. When there is no word \page, different strings are added to the file depending on whether it is the beginning of the file or otherwise.

```
#!/bin/ksh
#usage: script7.sh xxx.lst xxx.rtf

sed 's/\o14/\\page /' $1 |
awk '/\\page / {print $0}
!/\\page / {if(NR==1) \
{print "{\\rtf1\\ansi\\ansicpg1252\\deff0\\deflang1033 \
{\\fonttbl{\\f0\\fmodern\\fprq1\\fcharset0 Courier New;}} \
{\\colortbl \\red0\\green0\\blue0;} \
\\paperw15840\\paperh12240\\margl1440\\margr1440\\margt1440\\margb1440\\landscape \
\\viewkind4\\uc1\\pard\\ql\\fi0\\li0\\ri0\\sb0\\sa0\\sl-175 \\cf0\\f0\\fs14 \
\\hich\\af2\\dbch\\af31505\\loch\\f2 " $0}
else  {print  "\\par \\hich\\af2\\dbch\\af31505\\loch\\f2 " $0} }'  - > $2
echo ' '}'' >> $2
```

CONCLUSION

The shell scripting is a wonderful tool. SAS programmers working in Unix/Linux environment can utilize the tool and find interesting and efficient ways of dealing with text files such as LOG files and LST files created by SAS.

REFERENCES

Franklin, David. 2010. "Making an RTF file Out of a Text File, With SAS." *Proceedings of PharmaSUG2010, Paper CC14.*

CONTACT INFORMATION

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

Hengwei Liu
Daiichi Sankyo, Inc.
211 Mount Airy Road
Basking Ridge, NJ 07920
Hengwei_liu@yahoo.com

APPENDIX

Table 1 is a sample table in LST format.

XXXXXXXXXXXXXX, Inc. Page 1 of 20
Protocol: XXX-XXX-XXX Program Name: demog.sas
Interim Analysis Run datetime: <2019-03-31 12:43>

Table 14.1.2 Demographics and Baseline Characteristics
Safety Population

	Treatment A (N=xx)	Treatment B (N=xx)
Age		
n	xx	xx
Mean	xx.x	xx.x
Standard Deviation	xx.x	xx.x
Median	xx.x	xx.x
Min, Max	xx.x, xx.x	xx.x, xx.x
Race		
White	xx	xx
Black or African American	xx	xx
Asian	xx	xx
Other	xx	xx
Sex		
Male	xx	xx
Female	xx	xx

Note: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Program: /xxx/xxx/xxx/xxx/demog.sas

Table 1. A Sample Table in the LST Format