Paper 179-2011

Using SAS Macro Language and SAS® Pipe to Process State Health Care Survey Data

Xiaoting Qin, Cathy M. Bailey, Centers for Disease Control and Prevention (CDC), Atlanta, GA, US

ABSTRACT

States collect survey data continuously through the year using a standardized questionnaire from the Centers for Disease Control and Prevention (CDC). CDC receives data files from 40 states on an unscheduled basis. Monitoring each state's data collection by importing multiple ASCII files, converting ASCII files into SAS data sets, and compiling the SAS data sets for processing can be extremely time consuming and challenging for a SAS programmer. In this paper, we discuss a solution to automatically read 1-12 ASCII files from 40 state's folders on a network drive, convert multiple ASCII files to SAS data sets, and merge all of the SAS data sets into one aggregate SAS data set. We used Base SAS, the SAS macro language, and the Microsoft Windows operating system for the entire process. This method can be applied to other state-based survey data processing to efficiently control survey data quality.

INTRODUCTION

The Centers for Disease Control and Prevention (CDC) Behavioral Risk Factor Surveillance System (BRFSS) Asthma Call-back Survey is a cross-sectional telephone survey conducted by state health departments. A standardized questionnaire is used to determine the distribution of risk factors, behaviors, and health practices among BRFSS respondents with a diagnosis of having asthma at some point in their lives.

Forty states participated in the CDC BRFSS Asthma Call-back Survey in 2010. Although procedures were established to ensure data processing consistency, each state deviated from the procedures, thus creating data processing challenges. States collected data continuously throughout the year and they had the flexibility to submit data to CDC at anytime. This flexible data submission schedule made it time consuming to manually track the number of data files in each state's folder. Furthermore, data files included incomplete records that may or may not be updated and compiled into a later submission. To add additional complexity, subsequent file submissions could include or exclude previously submitted records. In this paper, we discuss how Base SAS and the SAS macro language were used to automatically check 40 folders on a network drive, monitor data collection activities, replace old records with updated records, and merge multiple SAS datasets into one aggregated SAS dataset for later data processing.

PROCESS

Step 1: Access Data File Information:

The DRIVE macro was defined to access the state's folder on the network drive and run the data processing steps state by state. The macro creates a two digit state name for all participating states, counts the total number of participating states, runs a DO loop to specify each state's folder on the network drive, and calls the READ_FILE and AGGREGATED macros to process the data state by state (Figure 1).

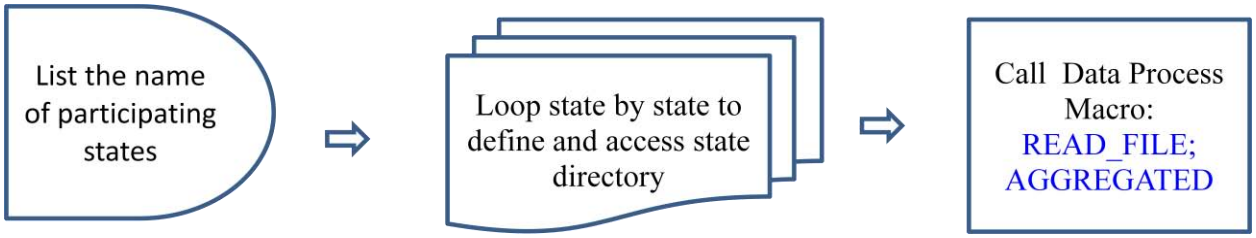

Figure 1. DRIVE Macro Operation Flow: Loop State by State to Process Data

The step one SAS sample source code:

```
%MACRO DRIVE;
%GLOBAL STATE MONTH;

 /* List the participating states */
%LET ST = AK AZ CA CO CT DC FL GA HI IA IL IN KS MA MD ME MI MS MT MO NE;
/* Count participating states */
%LET E = %EVAL(%SYSFUNC(COUNT(&ST,%STR( )))+1);

%DO A = 1 %TO &E;   /* loop state by state*/

/* Create state abbreviation FIPCODE and state name */
%LET STATE = %SCAN(&ST,&A);
%LET STFIPC = %SYSFUNC(STFIPS(&STATE));
%LET STSTR = %SYSFUNC(FIPSTATE(&STFIPC));
%LET STNAME = %SYSFUNC(FIPNAME(&STFIPC));

/* Define state folder */
LIBNAME &SRSTR ||&FILESRVR.\&STSTR\=;

 /* Macro to import and aggregated data file into SAS Data set*/
%READ_FILE;
%AGGREGATED;
%END;

%MEND;
```

Step 2: Import Files:

The READ_FILE macro imports each ASCII file, converts them to SAS data sets, and saves each SAS data set in the appropriate state folder. The challenge at this point is to determine the number of files sent by each state, capture the file name, and extract the file month from the file name. As part of the survey operation protocol, file naming conventions include three letters indicating the file month, and a two digit FIPS state code, regardless of how often the data were aggregated and sent to CDC. While executing the READ_FILE macro, SAS goes to the state folder, loops from Jan. to Dec. to import the ASCII files one-by-one and converts them into SAS datasets, if the month's data file exist, to create SAS monthly data sets (Figure 2). **Read File Name Using Pipe:** The macro obtains the names of

the SAS data sets to be read. It then brings the names of the data sets into a SAS data set by using the SAS unnamed pipe channel of communication. A pipe is a very useful device that allows two processes to communicate (Miller, 2004). SAS defines an unmanned pipe as a one way channel of communication between two processes where “one application can write data only to the pipe while the other application reads from it.” This means that a specialized application can be used to provide information to a SAS session or vice versa. This technique is described in the SAS Technical Document TS-581, which is available on the SAS Institute web site. The PIPE keyword is specified in the FILENAME statement followed by the operating system command in quotes. The operating system command produces a list of file names that exist in each state’s folder. The filename statement is then referenced in an INFILE statement in a DATA step to begin the process of reading the filenames.

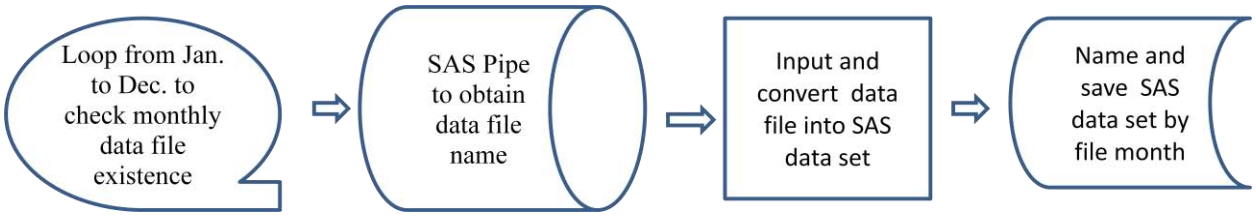

Figure 2. READ_FILE Macro Operation Flow: Read and Convert Multiple Data Files Into SAS Data Set in State Directory

The step two SAS sample sources code:

```
%MACRO READ_FILE;

/* Define three letters month name as macro variable */
%let M=JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC;

%DO J=1 %TO 12;
/* Read three letters month’s name one by one */
%LET MONTH=%SUBSTR(%STR(&M),(&J*4-3),3);
/* Name imported SAS dataset by file month name, save it in state folder */
%LET OUTFILE=&STSTR._&MONTH.;

/* Use Pipe keyword to list the data file name from Jan. to Dec.*/
/* Import existing monthly data file into SAS dataset */
FILENAME INDATA PIPE "DIR &FILESRVR.\&STSTR\&STSTR.&MONTH.*.DAT /B";

DATA &STSTR..&OUTFILE.;
     LENGTH FILE2READ $100;
     INFILE INDATA TRUNCOVER;
     INPUT DATA_FILE_NAME $30.;

     /* Link path and file name together */
     FILE2READ="&FILESRVR.\&STSTR\"|| DATA_FILE_NAME;
     INFILE DATAIN LRECL=1000 FILEVAR=FILE2READ END=EOF;
     INPUT;
```

```
        VARIABLE1
        VARIABLE2
        VARIABLE3
           .
           ;
RUN;
%END; /* End monthly loop*/

%MEND;  /* End Macro*/
```

Step 3: Aggregate Data:

The AGGREGATED macro merges the individual SAS monthly data sets into one SAS data set per state. Two main issues need to be solved in this step: update the data sets with latest records if duplicate, and merge multiple data sets into one aggregate dataset for each state. Since the number of data sets and file month name in each directory was unknown, the PIPE keyword was used to read the SAS data set name and the number of data sets to merge, which gets stored in a SAS data set. This SAS data set was later used extract the numerical file month name. Multiple datasets in each state's directory were merged into one aggregate state file. The merge sequence for the multiple SAS data sets was based on the SAS data set's file month (from Jan. to Dec) to ensure the later month's records will override the previous records, if duplicates exist (Figure 3).

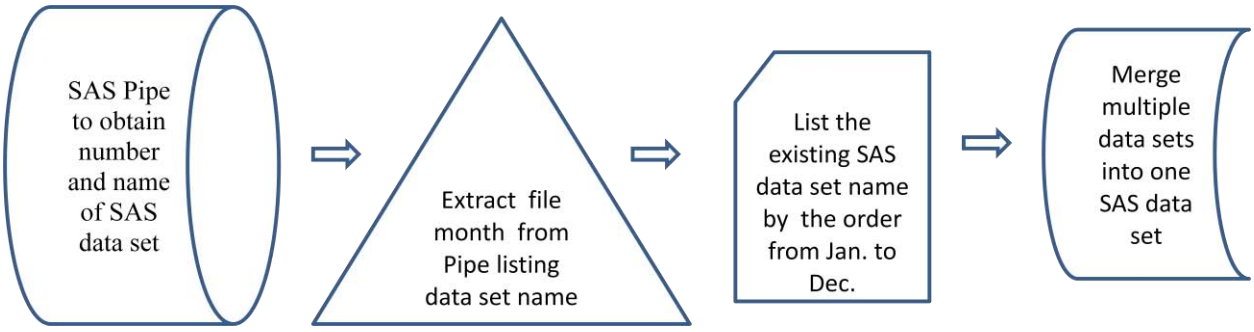

Figure 3. AGGREGATED Macro Operation Flow: Combined Multiple SAS File into One SAS Dataset

The step three SAS sample sources code:

```
%MACRO AGGREGATED;

%LET YR=2010;

/* List the SAS dataset name, get the number of SAS data files in state
folder*/
FILENAME INDATA PIPE "DIR DIR &FILESRVR.\&STSTR.\*.sas7bdat  /B";

/* Store the SAS dataset in a SAS data set*/
DATA FILE_LIST;
     LENGTH FNAME $100;
     INFILE INDATA TRUNCOVER;
     INPUT FNAME $100.;
```

```
/* Substract the SAS data set name,*/
     F_NAME=SUBSTR(FNAME,1,6);
/* Store the record number in a Macro variable*/
     CALL SYMPUT ('NUM_FILES', _N_);
/* Create SAS format date using SAS data set file month name */
     DATE_F='01'||TRIM(SUBSTR(FNAME,4,3))||=&YR=;
/*Extract the numerical file month name from SAS date*/
     MON=MONTH(INPUT(DATE_F,DATE9.));
     DROP FNAME DATE_F;
RUN;

/* Convert all F_NAME into Macro variable &ALLVAR*/
PROC SQL NOPRINT;
     SELECT F_NAME INTO :ALLFILE SEPARATED BY ' '
     FROM FILE_LIST
/* Order by dataset name by file month, from Jan. to Dec. (1-12) */
     ORDER BY MON;
QUIT;

/* Get the file name from FILE_LIST */
/* Merge multiple SAS file by the sequence from Jan. to Dec. */
%DO J=1 %TO &NUM_FILES.;
     DATA _NULL_;
/* Get the SAS file name one by one from FILE_LIST */
     SET FILE_LIST(OBS=&J);
 /* store the file name in Macro variable */
          CALL SYMPUT('FILEIN',F_NAME);
     RUN;
     PROC SORT DATA=&FILEIN.; /* Sort the SAS dataset */
          BY _STATE SEQNO;
     RUN;
%END;   /*End do loop*/

/* Merge the SAS data set, order by file numerical file month */
DATA &MAST_FILE.;
/* &ALLFILE is ordered from Jan ╥ Dec (1-12), later month records replace the
previous records if duplicate */
     MERGE &ALLFILE;
     BY _STATE SEQNO;
RUN;

%MEND;
```

CONCLUSION

Using SAS Pipe and the SAS macro language to automatically check folders on a network drive and read multiple data files from individual network folders has proven to be a very effective solution for survey data management, especially when the data submission rules were not strictly followed, and data file structures are not consistent. The SAS macro language proved an efficient means to process data state-by-state. The SAS Pipe channel of communicate is a powerful and efficient tool to use, to capture information about SAS and non SAS data files and file directory information. Automation of this tedious and error-prone process helped monitor data collection and process

data in timely manually. The procedures can be applied to any tasks which required consolidating multiple files from different data resources, where the data file name and number of files are unknown.

REFERENCES

Miller, Debbie (2004) “A Macro for Reading Multiple Text Files” Proceedings of the SAS Global 2004 Conference. http://www2.sas.com/proceedings/sugi29/057-29.pdf

SAS Institute Inc. (1998), Technical Support Document TS-581: Using FILEVAR= to Read Multiple External Files in a DATA Step, Cary, NC: SAS Institute Inc. http://support.sas.com/techsup/technote/ts581.pdf

Varney, Brian (2008) “Check out These Pipes: Using Microsoft Windows Commands from SAS®” Proceedings of the SAS Global 2008 Conference.
http://www2.sas.com/proceedings/forum2008/092-2008.pdf

Foty, Fuad (2009) “Clearing the Pipe: Overcoming System Limitations of the SAS® Pipe Channel of Communication” Northeast SAS User Group proceeding.
http://www.nesug.org/Proceedings/nesug09/ap/ap13.pdf

CONTACT INFORMATION

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

Name: Xiaoting Qin
Enterprise: Centers for Disease Control Prevention
Address: 4770 Buford High way, MS F-58
City, State ZIP: Chamblee, GA 30341
Work Phone: 770-488-0797
Fax: 770-488-1540
E-mail: CSK5@cdc.gov
Web: CDC.GOV