Paper P0-13

Customizing Proc Import Code

Carolyn D. Williams, UNC Highway Safety Research Center, Chapel Hill, NC

ABSTRACT

Proc import is a great SAS tool that I use often when converting raw data to SAS®. Usually SAS zips through the import procedure and the output is exactly what I expect. However occasionally there are glitches and 'gotcha' with numeric and character data types which can really make for a very bad day. The GUESSROWS= option value is valid for 1 to 32767 rows beyond that, the data type is pretty much set. What happens at observation number 82767 and SAS has defined a numeric storage location but the raw data has the value 302~40.33 or some other non numeric values for the next 200 records before it once again encounters valid numeric fields. This poster takes a serious and sometimes humorous walk down the yellow brick road to find the key to the import wizard's heart so that we might learn how to customize import statements and tame the bad (wicked) data

INTRODUCTION

In my capacity as a programmer Analyst for the Highway Safety Research Center, I am responsible for converting huge amounts of raw data into SAS format. This paper looks at the proc import procedure and some ways to modify the code provided to you by the import to correct errors listed in the program log.

THE PROBLEM

Data in format and types other than is given on the file layouts can present one of the most frustrating problems in programming. These data can be oracle dumps, delimited files, plain text files, access or excel files. They are usually accompanied by file descriptions and a short synopsis about the number of records on file, length, and in cases of delimited files, what character is used as the delimiter.

SAS import wizard is my tool of choice when processing delimited files. For small files less than 50,000 records, the procedure works perfect and very I quickly move on the next stage. The following shows the results of using the import wizard, when no column names are given.

```
/*********************************************************************
      *   PRODUCT:   SAS
      *   VERSION:   9.2
      *   CREATOR:   External File Interface
      *   DATE:      29JUN10
      *   DESC:      Generated SAS Data step Code
      *   TEMPLATE SOURCE:  (None Specified.)

*********************************************************************/
         data WORK.ACC08IL                                        ;
         %let _EFIERR_ = 0; /* set the ERROR detection macro variable */
         infile 'F:\HSISDATA\rawdata\Illin\2008\CrashExtract.txt'
delimiter = ',' MISSOVER DSD 235 lrecl=32767 ;

            informat VAR1 $14. ;
            informat VAR2 $9. ;
            informat VAR3 $5. ;
            informat VAR4 $4. ;
            informat VAR5 $4. ;

     format VAR1 $14. ;
            format VAR2 $9. ;
            format VAR3 $5. ;
            format VAR4 $4. ;
            format VAR5 $4. ;
            input
                   VAR1 $
                   VAR2 $
```

```
                        VAR3 $
                        VAR4 $
                        VAR5 $;
          if _ERROR_ then call symputx('_EFIERR_',1);  /* set ERROR detection
       macro variable */
                run;
```

Notice the results from the contents procedure below. The length of each character variable is the maximum length between the commas that separate each data items. Actual length in the file layout is what you really want.

Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Format	Informat
1	VAR1	Char	14	$14.	$14.
2	VAR2	Char	9	$9.	$9.
3	VAR3	Char	5	$5.	$5.
4	VAR4	Char	4	$4.	$4.
5	VAR5	Char	4	$4.	$4.
6	VAR6	Char	4	$4.	$4.
7	VAR7	Char	4	$4.	$4.
8	VAR8	Char	3	$3.	$3.
9	VAR9	Char	4	$4.	$4.

THE SOLUTION

With very little effort this code can be customize to output a file that has the lengths you need without having to use any SAS functions:

```
      data WORK.ACC08IL                                    ;
         %let _EFIERR_ = 0; /* set the ERROR detection macro variable */
         infile 'F:\HSISDATA\rawdata\Illin\2008\CrashExtract.txt' delimiter =
',' MISSOVER DSD 235 lrecl=32767 ;

            informat VAR1 $8. ;
            informat VAR2 $7;
            informat VAR3 $3.;
            informat VAR4 $2. ;
            informat VAR5 $2. ;

     format VAR1 $8. ;
            format VAR2 $7. ;
            format VAR3 $3. ;
            format VAR4 $2. ;
            format VAR5 $2. ;
            input
                  VAR1 $
                  VAR2 $
                  VAR3 $
                  VAR4 $
                  VAR5 $
;
    if _ERROR_ then call symputx('_EFIERR_',1);  /* set ERROR detection macro
variable */
         run;
```

If the file layout indicates the correct length then why not use the column name instead of the SAS default of var. In working with files with fewer than 25 variables I usually type them in. When customizing files with 40 or more variables, I open the documentation file into the SAS editor and use the mark block command. This feature is excellent when you want to cut and paste columns so now I have a means of acquiring the actual column name and length of each variable. This is not rocket science, just a wizard working its magic one line at a time as seen below:

```
      data WORK.ACC08IL                                     ;
         %let _EFIERR_ = 0; /* set the ERROR detection macro variable */
         infile 'F:\HSISDATA\rawdata\Illin\2008\CrashExtract.txt' delimiter =
',' MISSOVER DSD 235 lrecl=32767 ;

            informat CASENO $8. ;
            informat MILEPOST $7;
            informat ACCESS $3.;
            informat MONTH $2. ;
            informat DAY $2. ;

format CASENO $8.;
            format  MILEPOST $7. ;
            format ACCESS $3. ;
            format VMONTH $2. ;
            format DAY $2. ;
            input
                  CASENO $
                  MILEPOST $
                  ACCESS $
                  MONTH $
                  DAY $;
    if _ERROR_ then call symputx('_EFIERR_',1);  /* set ERROR detection macro
variable */         run;
```

CONCLUSION

Customizing Proc Import Code is not and easy solution if you have over fifty variables, but if this is data you receive on a regular basis, customization and adding macros can make your job a lot easier. SAS has products, one being, SAS Enterprise Guide that can handle problem data. The proc import is part of Base SAS and for those organizations that do not have the full suite of SAS products, customizing proc import is an easy alternative

CONTACT INFORMATION

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

Carolyn D. Williams
UNC Highway Safety Research Center

730 MLK Blvd, Suite 300
Chapel Hill, NC 27599
Work Phone: 919-962-8707
Fax: 919-962-8710
E-mail: cd_williams@unc.edu