Paper RF-01-2013

Getting *Wild* with Imports

Kathryn Schurr, M.S., Spectrum Health-Healthier Communities, Grand Rapids, MI
Jonathan Wiseman, Spectrum Health-Healthier Communities, Grand Rapids, MI

ABSTRACT

Ever have many similar datasets to import and not enough time to come up with a good macro to import and merge them all together? Using a *wildcard* import in SAS® v9.3 will solve all of your woes. This paper presents an efficient technique for importing multiple data files under one folder at the same time. Then it will further address the wildcard import as a SAS® Macro that can be used time and again for various folders containing differing data. This is a great tool for records kept electronically per facility, per student, or per patient; and especially when an extravagant query based data warehouse is unavailable.

INTRODUCTION

There are times when multiple files of the same type need to be imported and merged into a single SAS® dataset in order to be analyzed and summarized. When each of these files in question consists of the same variables and has the same layout, it is possible to import and merge all the files in one succinct datastep with one quick and easy INFILE statement. Known as a *wildcard import*, this process can save the analyst valuable time and effort.

THE EXAMPLE

Five friends each have a favorite baseball player who they keep various statistics on. These measures include their first and last name, what team they played on, which position, the number of games they participated in, their batting averages, the number of RBIs and Home Runs per season. Each fan keeps these measures recorded in a CSV file per player. The CSV files are updated during each season to reflect changes in the various measures. A single CSV file (opened using Excel) looks like the following:

	A	B	C	D	E	F
1	**Measure**	**2008 Season**	**2009 Season**	**2010 Season**	**2011 Season**	**2012 Season**
2	Last Name	Beltre	Beltre	Beltre	Beltre	
3	First Name	Adrian	Adrian	Adrian	Adrian	
4	Team	Seattle Mariners	Seattle Mariners	Boston Red Sox	Texas Rangers	
5	Position	3B	3B	3B	3B	
6	Games Played	143	111	154	124	
7	Batting Avg.	0.266	0.265	0.321	0.296	
8	RBI	77	44	102	105	
9	HR	25	8	28	32	
10						
11						
12						
13						

THE LONG WAY

Because each of the 5 friends is keeping separate files, importing the 5 different datasets can be accomplished using multiple data steps with INFILE statements and a final data step with a merge statement. The following code will import just one of the 5 different files. Similar code can be written to import the remaining four files; the dataset name and the file locations need to be individualized for each of the remaining files.

```
DATA A_Beltre;
   INFILE 'C:\Documents and
      Settings\Carl\Desktop\Baseball\A_Beltre.csv' DSD TRUNCOVER
      FIRSTOBS = 2;
   INPUT Measurement $ Season2008 $ Season2009 $ Season2010 $
      Season2011 $ Season2012 $;
RUN;
```

The statements and options used for importing the separate CSV files are the following:

INFILE: This specifies the file being read into SAS®.

DSD: This option allows SAS® to recognize the delimiter. Whenever the DSD option is used, two delimiters in a row signifies a missing value; the use of a delimiter inside of a quoted string is also ignored with this option.

TRUNCOVER: This option prevents SAS® from continuing onto the next line in the file whenever a short line is being read; this option will also fill in missing values where needed at the end of a data line.

FIRSTOBS = 2: This option tells SAS® to begin reading the data from the file at observation 2. This option is useful when unwanted variable names are included on the first line of the file being read in.

INPUT: This statement tells SAS® which variables are to be read and designates them as character or numeric values.

Once all 5 files have been imported and named, the next step is to combine all of the files into a single 'Players' dataset. The following code does this straightforwardly.

```
DATA Players;
   SET A_Beltre B_Posey D_Jeter J_Bruce M_Cabrera;
RUN;
```

Because each player's individual dataset has the same variables, a SET statement can be used. This will stack the datasets on top of one another creating a single dataset.

Although this method works and is easily implemented, the code can get rather tedious when there are a great many files to import at one time. What about 20? 80? 300? Macro's can import multiple files at once; however, when the number of files exceeds 20, macros themselves have their limitations. This is where a *Wildcard Import* can be extremely useful and time saving. Below, the *Wildcard Import* is explained.

THE *WILD* WAY

To avoid having to create multiple import statements in SAS® and combining all of the imported files together; a *wildcard* statement may be used to import the whole collection of CSV files and compile them into a single dataset. A *Wildcard* is signified when an asterisk character (*) is used in place of a file name, a portion of a file name, or a file location. SAS® recognizes an * as a *Wildcard* and this allows SAS® to read in multiple files. This is the technique for importing more than one file . An example of using the *Wildcard* in SAS® code is given below.

```
DATA Players;
   LENGTH MyInFile $400.;
   INFILE 'C:\Documents and Settings\Carl\Desktop\Baseball\*.csv' DSD
      TRUNCOVER FIRSTOBS = 2 FILENAME = MyInFile;
   INPUT Measurement $ Season2008 $ Season2009 $ Season2010 $
      Season2011 $ Season2012 $;
   EndFile = REVERSE(MyInFile);
   File = REVERSE(SUBSTR(EndFile,1,INDEX(EndFile,'\') - 1));
   KEEP Measurement Season2008 Season2009 Season2010 Season2011
      Season2012 File;
RUN;
```

The statements and options that are different when using a *Wildcard Import* than the ones used in the previous section are the following:

LENGTH: This file specifies the number of characters that the variable 'MyInFile' will have.

INFILE: This specifies the file being read into SAS®. By using the '*.csv' at the end of INFILE file specification, SAS® will import all files within the given directory that are of a CSV type, regardless of the name of the file.

FILENAME = MyInFile: This option tells SAS® to record each of the filenames that are being imported as a SAS® variable.

EndFile: By using the REVERSE() function in SAS®, the file name and pathway is recorded backwards. This will be useful in the next line of code.

File: This formula uses the INDEX () function in SAS® to specify a portion of the filename up to a certain character. In this case, we want to grab only the file name, not the directory or full path name indicating where this file is located, so the character value used in the INDEX function is '\'. Once the index character is recognized, we want to take a substring of the full path name that is recorded in the *EndFile* variable up to the first '\'. Because we reversed the string originally, the substring will begin at the first character and run until '\' is reached. We will then apply the REVERSE () function again which will reverse the characters of the filename so that they are in their original position. The table below shows each step:

Full Path Name	C:\Documents and Settings\Carl\Desktop\Baseball\A_Beltre.csv
Reversed Path Name	vsc.ertleB_A\llabesaB\potkseD\larC\sgnitteS dna stnemucoD\:C
Indexed Substring	vsc.ertleB_A
Reversal of the Substring (Final File Name)	A_Beltre.csv

KEEP: This statement specifies which variables to keep in the final dataset. Because there is no use for the Full Path Name or the Reversed Path Name, those variables are not included in this statement.

Because each player's individual dataset has the same variables, the use of a wildcard statement is possible. Even if the variables in each dataset were named differently (e.g. one file had 'S12' instead of 'Season2012'), by specifying an input statement all imported variables will have the same variable name.

CONCLUSION

By using the *Wildcard Import* as specified above, time and effort may be saved when there are numerous files to be imported and combined. This method is particularly useful when there are records that are kept electronically per patient, per student, or per location. The previous example used only CSV files; however, this technique may be modified to accommodate any data format.

REFERENCES

Delwiche, Lora D., and Susan J. Slaughter. *The Little SAS Book*. Third ed. Cary NC: SAS Publishing, 2003. N. pag. Print.

First, Steven. "The SAS INFILE and FILE Statements." SAS Global Forum 2008. Systems Seminar Consultants, 2008. Web. 12 Apr. 2013. Path: http://www2.sas.com/proceedings/forum2008/TOC.html.

CONTACT INFORMATION

Your comments and questions are much appreciated. Contact the authors at:

Kathryn Schurr, M.S.
Statistical Database Analyst
Spectrum Health Corporate
Healthier Communities
665 Seward Avenue NW, Suite 110
Grand Rapids, MI 49504
Work Phone: 616.391.2983
Work E-Mail: kathryn.schurr@spectrumhealth.org