Paper CT-17

What's in a FILENAME?

Heidi Markovitz, Federal Reserve Board of Governors, Washington, DC

ABSTRACT AND INTRODUCTION

The FILENAME statement is an old standard that assigns a nickname or file handle to a non-SAS® data location. It is most commonly used to point to a single flat file. However, for many years it has been able to act on an explicit list of files or a location that contains an unknown batch of files. This presentation will show how to use these capabilities, along with the INFILE statement in a single DATA step, to combine a list of files, explore all the files in a directory (folder), or build a list of files of a certain type.

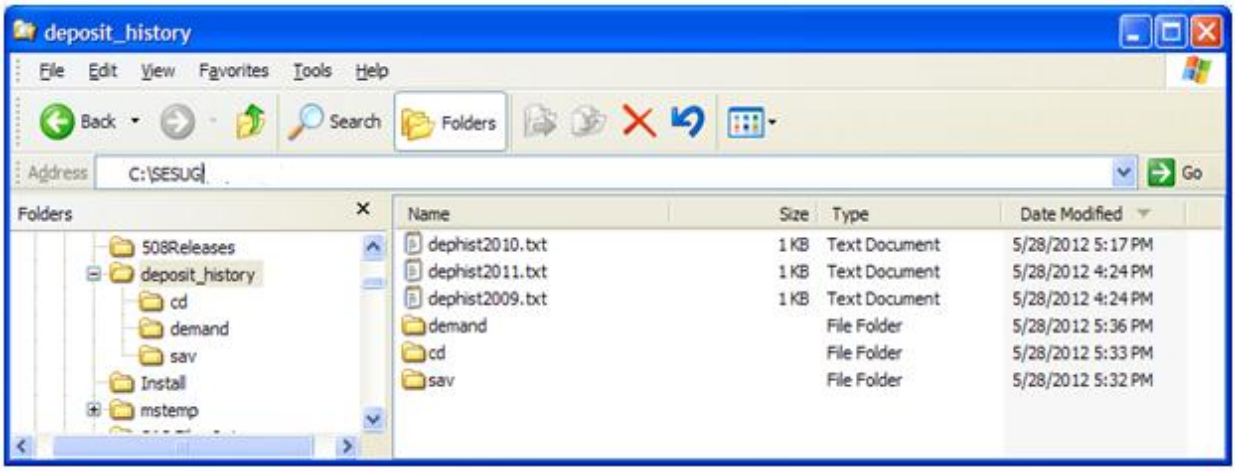
ORIGIN

The FILENAME statement originated in Version 5. It was invented to make it easier for SAS programmers to run the same program on different hardware platforms. Incompatible file naming rules across platforms meant that a program that referred to a file by its complete name would need to be modified before it could be run on another platform. FILENAME allowed a programmer to associate a short nickname with the actual file at the beginning of the code and subsequently use only the nickname.

Starting in Version 6, FILENAME could assign a single file handle to multiple files. A SAS program would treat the files as though they had been concatenated, or pasted together, one after the other. The material that follows shows why and how to use this feature.

SAMPLE DATA

The data files used in this presentation were inspired by data we use at the Federal Reserve Board. They are periodic files containing bogus deposit amounts for individual banks on certain dates. For illustration purposes, they are organized into several directories. An excerpt of the data file tree is shown below.



And here is the content of one of the input text files, dephist10.txt.

```
bank_id  date_cymd avg_deposits
   1010  20090826   1515.00
   1020  20090816   1530.00
   1030  20090806   1545.00
   1040  20090727   1560.00
   1050  20090717   1575.00
   1060  20090707   1590.00
   1070  20090627   1605.00
   1080  20090617   1620.00
   1090  20090607   1635.00
   1100  20090528   1650.00
```

NOTE: The techniques shown in this paper have been tested on Microsoft Windows and Unix/Linux platforms. The

examples in the body of the paper are from MS/Windows.

BASIC FILENAME

Here are examples of what is still the most common use - giving a short nickname to a single input or output file.

Microsoft Windows Example:
```
FILENAME goodata     "C:\user data\dephist2010.txt";
```

Unix/Linux Example:
```
FILENAME goodata     "/user data/dephist2010.txt";
```

Both assign goodata as a reference to file dephist2010.txt in folder/directory user data.

SAMPLE DATA STEP

Either FILENAME statement will work to enable the following DATA step to process input file dephist2010.txt.

```
1. DATA alldep;
2. INFILE goodata TRUNCOVER;
3. INPUT bank_id  ??
         date     ??
         avg_deps ??;
4. IF NOT MISSING(bank_id)
      THEN OUTPUT;
5. ELSE PUT 'Record ' _N_ 'is NOT numeric: ' _infile_;
6. RUN;
```

Explanation of selected numbered statements:
1-States that the DATA step will create temporary dataset alldep in the default WORK directory.
2-Use the file referenced by goodata in an earlier FILENAME statement as input. In later examples, "goodata" will be replaced by filerefs defined in other FILENAME statements. The TRUNCOVER option allows input records that are incomplete (truncated) to be processed.
3-For each record in goodata, read the first three space separated fields. Assign the values from those fields to numeric variables bank_id, date, and avg_deps, respectively. The "??" prevents SAS from writing automatic warning messages if any of the input fields are not numeric.
4-If the bank_id is numeric, output a record to new SAS data set alldep.
5-Otherwise (the bank_id is not OK),report on the bad value.

MULTIPLE NAMED FILES

There may be several files containing similar data that need to be combined. In this example, three files contain deposit data for different years. To combine them into one SAS dataset, start with this FILENAME statement. It assigns a single file reference, byname, to three files.

```
FILENAME byname (
   'C:\SESUG\dephist2009.txt'
   'C:\SESUG\dephist2010.txt'
   'C:\SESUG\dephist2011.txt'
);
```

Next, change the fileref in the sample data step above, to byname.
```
INFILE byname TRUNCOVER;
```

Last, submit the FILENAME statement and DATA step.

The INPUT statement will read file dephist2009.txt, from the first to the last record, then read all the records in dephist2010.txt, and finally read dephist2011.txt. The DATA step will produce SAS dataset alldep containing three variables (bank_id, date, and avg_deps) from each record with a numeric bank_id in all the input files associated with file reference byname.

Here is the part of the log showing the characteristics of each physical file specified in the FILENAME statement,

followed by the number of records read from each and the total observations in the resulting SAS dataset.

Partial log from Multiple Named Files:

```
146  RUN;

NOTE: The infile BYNAME is:
      File Name=C:\SESUG\dephist2009.txt,

      File List=('C:\SESUG\dephist2009.txt'
      'C:\SESUG\dephist2010.txt' 'C:\SESUG\dephist2011.txt'),
      RECFM=V,LRECL=256

NOTE: The infile BYNAME is:
      File Name=C:\SESUG\dephist2010.txt,

      File List=('C:\SESUG\dephist2009.txt'
      'C:\SESUG\dephist2010.txt' 'C:\SESUG\dephist2011.txt'),
      RECFM=V,LRECL=256

NOTE: The infile BYNAME is:
      File Name=C:\SESUG\dephist2011.txt,

      File List=('C:\SESUG\dephist2009.txt'
      'C:\SESUG\dephist2010.txt' 'C:\SESUG\dephist2011.txt'),
      RECFM=V,LRECL=256

NOTE: 3 lines were written to file PRINT.
NOTE: 11 records were read from the infile BYNAME.
      The minimum record length was 27.
      The maximum record length was 31.
NOTE: 9 records were read from the infile BYNAME.
      The minimum record length was 27.
      The maximum record length was 31.
NOTE: 10 records were read from the infile BYNAME.
      The minimum record length was 27.
      The maximum record length was 31.
NOTE: The data set WORK.ALLDEP has 27 observations and 3 variables.
NOTE: DATA statement used (Total process time):
      real time           0.09 seconds
      cpu time            0.03 seconds
```

The output below (which in real life may be in a .lst file or at the end of the log), reports on the records from the flat files that had non-numeric bank_ids. In this case, all the reported records were column headers for the input files.

```
      The SAS System           15:49 Thursday, May 31, 2012   1
Record 1  is NOT numeric: bank_id  date_cymd avg_deposits
Record 12  is NOT numeric: bank_id  date_cymd avg_deposits
Record 21  is NOT numeric: bank_id  date_cymd avg_deposits
```

PARTIALLY SPECIFIED FILES

What if the files to concatenate are too many to list individually? That was the situation that motivated me to do this presentation. There were quarterly files from more than 20 years and no one was sure what any of their layouts were. I needed to quickly get an idea of their contents. Here is a simplified adaptation of the directory and file organization:

```
SESUG\cd:
    cddeps2011
SESUG\demand:
    demand_NovDec2011
SESUG\sav:
    savdeps_Aug2009
    savdeps_Jul2009
    savdeps_Jun2009
    savdeps_May2009
```

The following FILENAME statement enables the same DATA step from earlier in the paper to read all the files in subdirectories sav, cd, and demand of parent directory C:\SESUG, using file reference bydir.

```
FILENAME bydir (
   'C:\SESUG\sav\*'
   'C:\SESUG\cd\*'
   'C:\SESUG\demand\*'
   );
```

Note that the path of each directory (folder) is spelled out (e.g. C:\SESUG\sav), but the names of the files are indicated by the wildcard character, "*" (asterisk). The folders will be processed as ordered on the FILENAME. The text files within each folder will be read alphabetically, by name.

To proceed, change the file reference in the old DATA step to bydir, like this.

```
INFILE bydir TRUNCOVER;
```

Submit the FILENAME statement and revised DATA step to create a single SAS dataset with each observation containing three variables for each record in all the input text files. Again, input records with non-numeric bank_ids will be omitted from the new dataset.

The log shows the names and other traits for each input file, even though they were not explicitly enumerated on the FILENAME statement. It continues with the number of records read from each of the files and the total number of observations output. And the output report lists all the input records that were rejected for non-numeric bank_ids.

Selected excerpts from log for Partially Specified Files:

```
187  DATA alldep;
188     FILE PRINT;
189     INFILE bydir      TRUNCOVER;
----skipping----

NOTE: The infile BYDIR is:
      File Name=C:\SESUG\sav\savdeps_Aug2009,

      File List=('C:\SESUG\sav\*' 'C:\SESUG\cd\*'  'C:\SESUG\demand\*'),
      RECFM=V,LRECL=256

NOTE: The infile BYDIR is:
      File Name=C:\SESUG\sav\savdeps_Jul2009,

      File List=('C:\SESUG\sav\*' 'C:\SESUG\cd\*'  'C:\SESUG\demand\*'),
      RECFM=V,LRECL=256
----skipping----

NOTE: The infile BYDIR is:
      File Name=C:\SESUG\demand\demand_NovDec2011,

      File List=('C:\SESUG\sav\*' 'C:\SESUG\cd\*'  'C:\SESUG\demand\*'),
      RECFM=V,LRECL=256

NOTE: 6 lines were written to file PRINT.
NOTE: 4 records were read from the infile BYDIR.
      The minimum record length was 27.
      The maximum record length was 31.
NOTE: 4 records were read from the infile BYDIR.
      The minimum record length was 27.
      The maximum record length was 31.
----skipping----

NOTE: 9 records were read from the infile BYDIR.
      The minimum record length was 27.
      The maximum record length was 31.
NOTE: The data set WORK.ALLDEP has 27 observations and 3 variables.
```

Output report from Partially Specified Files:

```
      The SAS System          15:49 Thursday, May 31, 2012   2

   Record 1  is NOT numeric: bank_id  date_cymd avg_deposits
   Record 5  is NOT numeric: bank_id  date_cymd avg_deposits
   Record 9  is NOT numeric: bank_id  date_cymd avg_deposits
   Record 13  is NOT numeric: bank_id  date_cymd avg_deposits
   Record 15  is NOT numeric: bank_id  date_cymd avg_deposits
   Record 25  is NOT numeric: bank_id  date_cymd avg_deposits
```

If hundreds or thousands of files are processed, the log will be quite long, but the information about each file is very useful for debugging.

MACRO-GENERATED FILE LIST

Extending the earlier case of too many files to list individually, a SAS Macro can generate a subset of a long list of files. A practical use would be choosing the files from the last day of each month from several years of daily files.

This simplified example makes and uses a SAS Macro to build a list of files based on a predictable naming pattern. As you can see in this list, the SESUG directory contains a series of files named for each year from 1995 to 2012.

```
SESUG:
    dephist1995
    dephist1996
    ----skipping----
    dephist2009
    dephist2010
    dephist2011
    dephist2012
```

To combine the files for 2009-2011, start by defining SAS Macro filelist.

```
%MACRO filelist(startpath);
  FILENAME bymacro (
       %DO fileyear=2009 %TO 2011;
    "&startpath\dephist&fileyear"
       %END;
    );
%MEND filelist;
```

When invoked, this macro builds a FILENAME statement that assigns fileref bymacro to files named dephist&fileyear in the directory specified by the startpath parameter. &fileyear must be 2009, 2010, or 2011. If the macro is applied to the directory shown above, the FILENAME statement will refer to dephist2009, dephist2010, and dephist2011.

To combine the three annual files selected by the macro into one SAS dataset, change the fileref on the INFILE statement in the DATA step to the one defined in the macro, bymacro.

```
INFILE bymacro TRUNCOVER;
```

Then submit the following.

```
 1. OPTIONS MPRINT SYMBOLGEN;
 2. %filelist(C:\SESUG);

 3. DATA alldep;
 4.    INFILE bymacro TRUNCOVER;
       ----skipping----
 5. RUN;
```

Explanation of selected numbered statements:

1-Sets system options that will show the SAS statements generated by the macro and the values of macro variables referenced in the program. Since these options can add a lot to the log, it may be a good idea to remove this statement once the program is working correctly.

2-Invokes macro filelist with parameter C:\SESUG, the name of the directory containing the files to be processed.

3-The start of the same DATA step used throughout this paper.

The SYMBOLGEN output in the log shows the values of each macro variable as macro filelist executes. Macro variable startpath contains the name of the directory being evaluated. It is C:\SESUG, the value passed to filelist when it was called. Fileyear contains the year part of each file name. Its value is controlled during execution by the iterative %DO loop in the macro.

SYMBOLGEN output from log for Macro-Generated File List:

```
SYMBOLGEN:  Macro variable STARTPATH resolves to C:\SESUG
SYMBOLGEN:  Macro variable FILEYEAR resolves to 2009
SYMBOLGEN:  Macro variable STARTPATH resolves to C:\SESUG
SYMBOLGEN:  Macro variable FILEYEAR resolves to 2010
SYMBOLGEN:  Macro variable STARTPATH resolves to C:\SESUG
SYMBOLGEN:  Macro variable FILEYEAR resolves to 2011
```

The MPRINT output in the log is the result of running macro filelist: a FILENAME statement almost identical to the one coded manually in Multiple Named Files, above. It assigns file reference bymacro to the concatenation of files dephist2009, dephist2010, and dephist2011 in folder C:\SESUG.

MPRINT output from log for Macro-Generated File List:

```
MPRINT(FILELIST):   FILENAME bymacro (
"C:\SESUG\dephist2009"
"C:\SESUG\dephist2010"
"C:\SESUG\dephist2011" );
```

The rest of the log and the report output are the same as the Multiple Named Files example, except that the infile name is bymacro, not byname. Here are short excerpts from the later log.

```
NOTE: The infile BYMACRO is:
      File Name=C:\SESUG\dephist2009,

      File List=('C:\SESUG\dephist2009' 'C:\SESUG\dephist2010'
      'C:\SESUG\dephist2011'), RECFM=V,LRECL=256

NOTE: The infile BYMACRO is:
      File Name=C:\SESUG\dephist2010,

      File List=('C:\SESUG\dephist2009' 'C:\SESUG\dephist2010'
      'C:\SESUG\dephist2011'), RECFM=V,LRECL=256
----skipping----
NOTE: 3 lines were written to file PRINT.
NOTE: 11 records were read from the infile BYMACRO.
      The minimum record length was 27.
      The maximum record length was 31.
----skipping----
NOTE: The data set WORK.ALLDEP has 27 observations and 3 variables.
```

FILENAME OPTION OF INFILE STATEMENT

Writing only one DATA step to read many files is very useful, but it introduces the problem of locating peculiarities in the data. In the Partially Specified Files example, six files are processed. The log names each file, the order in which it was read, and how many records it had. But the output report which lists invalid input data, shows the sequence number of the problem record among all records in all files, making it tough to pinpoint which file contained the bad data. Fortunately, the FILENAME option can be added to the INFILE statement in the DATA step to help solve this problem. It provides a temporary variable containing the name of the file from which the current record was read. The variable value is automatically updated as one input file finishes and the next begins.

To demonstrate, here is a slightly modified version of our habitual DATA step. It selectively sums only the deposits from input files that include the year 2011 in their names. After all the files represented by fileref byname have been read, it prints the 2011 total.

```
1.  DATA alldep;
2.    LENGTH thisfile    $ 100;
3.    IF enddeps    THEN
4.      PUT '2011 Deposits:'
5.          totdep_2011 DOLLAR11.2;
6.    INFILE byname TRUNCOVER END=enddeps
7.      FILENAME=thisfile;
8.    INPUT bank_id  ??
            date     ??
            avg_deps ??;
9.    IF INDEX(thisfile, '2011')
10.     THEN totdep_2011 + avg_deps;
11. RUN;
```

Explanation of selected numbered statements:

2-The LENGTH statement makes variable thisfile large enough to contain the complete name of an input file because SAS will insert the complete path of the file into the variable even if the FILENAME statement contained an abbreviated name. In the Named Files example, thisfile would contain "C:\SESUG\dephistnnnn.txt" not "dephistnnnn.txt."

3-5-After the last input record in the last file is read, total deposits for 2011 will be reported. See the END= option on the INFILE statement.

6-The END= option on the INFILE statement creates variable enddeps that will contain 0 (zero) until the last record in the last file is read, when will be set to 1 (one).

7-The FILENAME= option on the INFILE statement assigns variable thisfile to be set to the name of the physical file being read.

9-10-Making use of variable thisfile defined by the FILENAME= option, SAS will check the name of the file currently being read. If the name contains "2011," SAS will add avg_deps from each record in the file to the total it is accumulating. In the Named Files example, it would contain the total from dephist2011.txt.

Output report from FILENAME Option of INFILE Statement:

```
The SAS System           12:54 Saturday, June 2, 2012   1

2011 Deposits:  $14,647.50
```

The same technique could be used to customize error-checking, based on the name of the input file.

CONCLUSION

The SAS FILENAME statement provides a way to process one or many files as a single stream of data. It is particularly flexible in working with large series of files that need to be processed together.

ACKNOWLEDGEMENTS

Thanks to Bruce Gilsen at the Federal Reserve Board of Governors for his constant encouragement and input. Thank you Joe Kelley, at the University of Georgia for helping with the historical background. My appreciation to Susan Dimmick, for her considered editing

CONTACT

Heidi Markovitz, Sr. Research Systems Analyst
Federal Reserve Board of Governors
(W) (202) 452-2724
(C) (305) 803-8407
HeidiM@Simply-Systems.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.
Other brand and product names are trademarks of their respective companies.