Tips for using the IMPORT procedure to read files that contain delimiters

6

By Amber Elam on SAS Users | October 20, 2017

Topics | Programming Tips

Reading an external file that contains delimiters (commas, tabs, or other characters such as a pipe character or an exclamation point) is easy when you use the IMPORT procedure. It's easy in that variable names are on row 1, the data starts on row 2, and the first 20 rows are a good sample of your data. Unfortunately, most delimited files are not created with those restrictions in mind. So how do you read files that do not follow those restrictions?

You can still use PROC IMPORT to read the comma-, tab-, or otherwise-delimited files. However, depending

Note: PROC IMPORT is available only for use in the Microsoft Windows, UNIX, or Linux operating environments.

Peter Styliadis, a trainer with SAS, shows how to use PROC IMPORT to read CSV data in this tutorial.

The following sections explain four different scenarios for using PROC IMPORT to read text files with delimited values.

Scenario 1: Variable names on row 1, values beginning row 2

In this scenario, I use PROC IMPORT to read a comma-delimited file that has variable names on row 1 and data starting on row 2, as shown below:

```
proc import datafile='c:\temp\classdata.csv'
out=class dbms=csv replace;
run;
```

When I submit this code, the following message appears in my SAS® log:

```
NOTE: Invalid data for Age in line 28 9-10.
RULE:     ----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8----+---
28        Janet,F,NA,62.5,112.5 21
Name=Janet Sex=F Age=. Height=62.5 Weight=112.5 _ERROR_=1 _N_=27
NOTE: 38 records were read from the infile 'c:\temp\classdata.csv'.
      The minimum record length was 17.
      The maximum record length was 21.
NOTE: The data set WORK.CLASS has 38 observations and 5 variables.
```

In this situation, how do you prevent the `Invalid Data` message in the SAS log?

By default, SAS scans the first 20 rows to determine variable attributes (type and length) when it reads a comma-, tab-, or otherwise-delimited file. Beginning in SAS® 9.1, a new statement (GUESSINGROWS=) is available in PROC IMPORT that enables you to tell SAS how many rows you want it to scan in order to determine variable attributes. In SAS 9.1 and SAS® 9.2, the GUESSINGROWS= value can range from 1 to 32767. Beginning in SAS® 9.3, the GUESSINGROWS= value can range from 1 to 2147483647. Keep in mind that the more rows you scan, the longer it takes for the PROC IMPORT to run.

The following program illustrates the use of the GUESSINGROWS= statement in PROC IMPORT:

```
proc import datafile='c:\temp\classdata.csv' out=class
               dbms=csv replace;
guessingrows=100;
run;
```

The example above includes the statement GUESSINGROWS=100, which instructs SAS to scan the first 100 rows of the external

BLOGS

Scenario 2: Variable names and data begin "later" than row 1

In this scenario, my delimited file has the variable names on row 4 and the data starts on row 5. When you use PROC IMPORT, you can specify the record number at which SAS should begin reading. Although you can specify which record to start with in PROC IMPORT, you cannot extract the variable names from any other row except the first row of an external file that is comma-, tab-, or an otherwise-delimited.

Then how do you program PROC IMPORT so that it begins reading from a specified row?

To do that, you need to allow SAS to assign the variable names in the form VARx (where x is a sequential number). The following code illustrates how you can skip the first rows of data and start reading from row 4 by allowing SAS to assign the variable names:

```
proc import datafile='c:\temp\class.csv' out=class
  dbms=csv replace;
  getnames=no;
  datarow=4;
run;
```

Scenario 3: Read a "section" of data from the middle of the file

In this scenario, I want to read only records 6–15 (inclusive) in the delimited file. So the question here is how can you set PROC IMPORT to read just a section of a delimited file?

To do that, you need to use the OBS= option before you execute PROC IMPORT and use the DATAROW= option within PROC IMPORT.

The following example reads the middle ten rows of a CSV file, starting at row 6:

```
proc import out=work.test2
            datafile= "c:\temp\class.csv"
            dbms=csv replace;
            getnames=yes;
            datarow=6;
run;

options obs=max;
run;
```

Notice that I reset the OBS= option to MAX after the IMPORT procedure to ensure that any code that I run after the procedure processes all observations.

Scenario 4: PROC IMPORT produces more observations than you expect

In this scenario, I again use PROC IMPORT to read my external file. However, I receive more observations in my SAS data set than there are data rows in my delimited file. The external file looks fine when it is opened with Microsoft Excel. However, when I use Microsoft Windows Notepad or TextPad to view some records, my data spans multiple rows for values that are enclosed in quotation marks. Here is a snapshot of what the file looks like in both Microsoft Excel and TextPad, respectively:

The question for this scenario is how can I use PROC IMPORT to read this data so that the observations in my SAS data set match the number of rows in my delimited file?

In this case, the external file contains embedded carriage return (CR) and line feed (LF) characters in the middle of the data value within a quoted string. The CRLF is an end-of-record marker, so the remaining text in the string becomes the next record. Here are the results from reading the CSV file that is illustrated in the Excel and TextPad files that are shown earlier:

A sample program that removes a CRLF character (as long as it is part of a quoted text string) is available in SAS Note 26065, "Remove carriage return and line feed characters within quoted strings."

After you run the code (from the Full Code tab) in SAS Note 26065 to pre-process the external file and remove the erroneous CR/LF characters, you should be able to use PROC IMPORT to read the external file with no problems.

For more information about PROC IMPORT, see "Chapter 35, The IMPORT Procedure" in the *Base SAS® 9.4 Procedures Guide, Seventh Edition*.

Tags

sas programming | SAS Programmers | Problem Solvers

Share

ABOUT AUTHOR

Amber Elam is a Technical Support Analyst in the Foundation SAS group in Technical Support. She has worked in Technical Support at SAS for 26 years and provides general support for the DATA step, macro, and Base procedures. Amber has her SAS Certification in Base Programming for SAS 9.

6 COMMENTS

Tom Abernathy on May 8, 2021 12:53 am

@ "PRASHANT CHEGOOR" - You can use longer LRECL by creating a FILEREF. But PROC IMPORT will only use the first 32,767 bytes of the header row when looking for header text to use to guess how to name the variables.

The trick in SAS Note 26065 does not remove CR and/or LF characters. Since it uses SHAREBUFFER option it cannot remove characters only replace them. Note this also means it changes the file so make a back-up before running that code. Or modify it to write the corrected version to a new file, in which case you could have it remove either the CR or LF or both.

To really get a fix to scenario 4 follow this link to up-vote the SASware Ballot Idea to have SAS enhance their support for delimited files.
https://communities.sas.com/t5/SASware-Ballot-Ideas/Enhancements-to-INFILE-FILE-to-handle-delimited-file-variations/idi-p/435977

Reply ›

Tabi on March 3, 2021 2:19 pm

I am trying to read in a csv file but there is commas inside some of the values of a variable without surrounded by quotes. would you please advise me how to solve this issue using proc import or using infile. thanks in advance!

Reply ›

BLOGS

Please open up a track with SAS Technical Support to address your question. Here is the link to opening up a track:

https://support.sas.com/en/technical-support/contact-sas.html

Reply ›

Mahan on September 17, 2020 3:10 am

Hello everyone, I'm trying to import a CSV file into SAS and I'm getting the following error with not many details:

```
ERROR: Import unsuccessful. See SAS Log for details.
NOTE: The SAS System stopped processing this step because of errors.
NOTE: PROCEDURE IMPORT used (Total process time):
real time 0.02 seconds
cpu time 0.03 seconds
```

My code:

```
%let path=G:\Files;
options validvarname=v7;

proc import datafile="&path\My.csv"
out=My_File
dbms=csv;
guessingrows=max;
run;
```

Reply ›

Unnamed Voice on July 9, 2019 8:09 pm

BLOGS

Thank you for scenario 4. I was not able to google my way to the solution, but found your blog which pointed to SAS Note 26065.

Reply ›

PRASHANT CHEGOOR on October 20, 2017 4:23 pm

Can we change the LRECL value to beyond 32K for the file being read using a Filename Statement and PROC IMPORT ? Or is it only restricted to 32,767?

Reply ›

LEAVE A REPLY

Your Comment

Your Name

Your Email

BLOGS

☐ Save my name, email, and website in this browser for the next time I comment.

Post Comment

This site uses Akismet to reduce spam. Learn how your comment data is processed.