

Post STA 216 Reintroduction to SAS

Please note that I only use SAS Windowing Environment (aka Display Manager), as opposed to SAS Enterprise Guide (EG). You are welcome to use EG if you like, but I have never used EG, and I will demonstrate all analyses this semester in SAS Windowing Environment. Also, there are many other ways to accomplish the following tasks, but this is just how I do things. If you have another way to get the job done, please feel free, as long as you can document it by pasting your code into your assignments/labs.

General reminder: If you are writing SAS code, and you are not getting the results you want, the very first thing you should do is look at the SAS log file. While the error messages are not always easy to understand, at the very least, SAS tells you where the error is.

There are three main features of SAS you need to be able to use:

1. Getting data into SAS from an external source, typically Excel or CSV files
2. Using a data step to create new SAS data sets from existing SAS data sets, including creating new variables.
3. SAS procedures (proc's)

1. Getting data into SAS

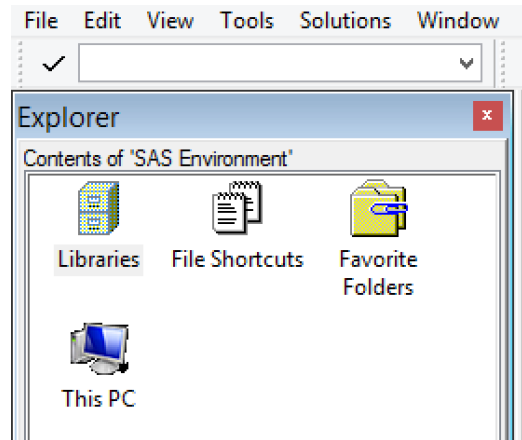
Before you bring data into SAS, I recommend creating a permanent SAS library. This will prevent you from having to reimport the data set each time you want to use it. A library is just a pointer, to tell SAS what folder to store your new SAS data set (suffix .sas7bdat). Once you have imported the data into SAS, then you can simply refer to it using

libraryname.datasetname

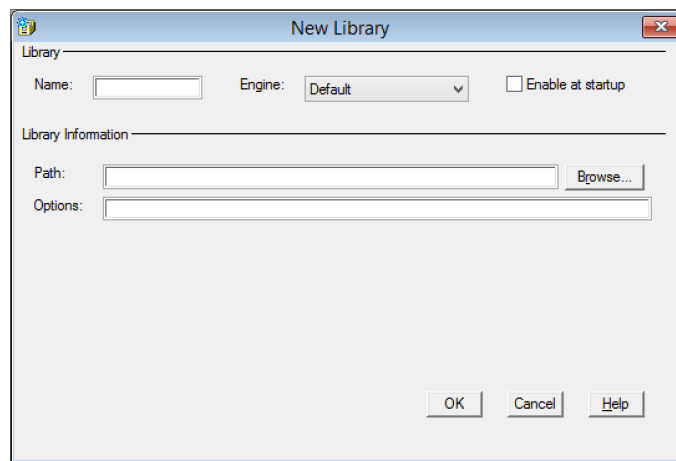
There is already a temporary library called "work", which you can use, but then you would need to reimport the data each time you open SAS.

While you can write code to create a permanent SAS library, I prefer to use point and click. I should mention that if you are using SAS on the blade, you may encounter issues with creating libraries. If that happens, I recommend just using SAS on the network.

- Decide on a folder where you will keep all of your documents for this class. This could be on your personal network space, typically called the Z or (formerly N?) drive.
- Open SAS and click on Explorer near the bottom left corner of the SAS window. You should see a library icon in the upper left, and SINGLE click on it.



- Click on File, New and then you should see

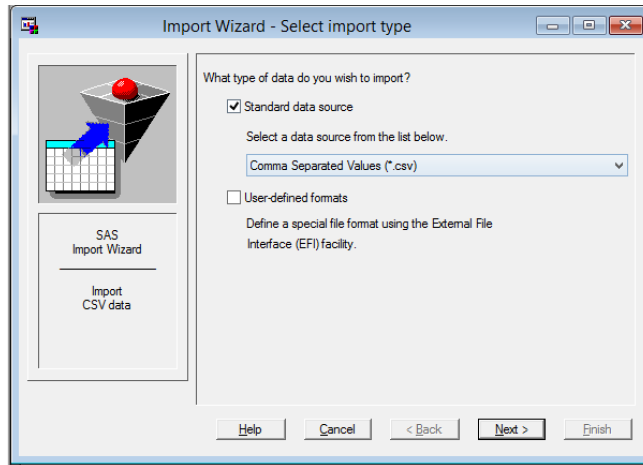


- Type your library name, such as sta310, all as one word (no spaces).
- Click on enable at startup.
- Specify the path to your folder where you will keep all your STA 310 data sets.
- Click OK
- Now if you double click on the library icon, you should see all of your SAS libraries, which include some built in ones to ignore, but you should see your new sta310 library, along with the Work library.

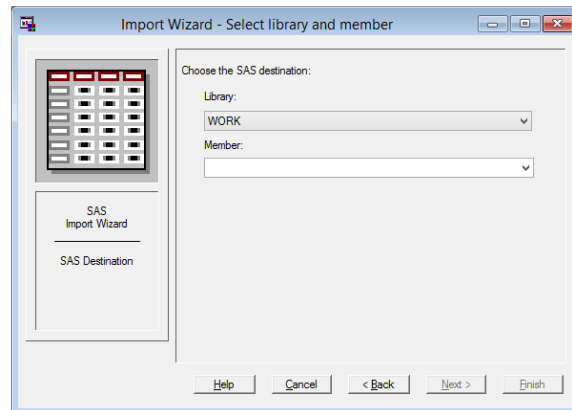
Now we are ready to import data into SAS. I prefer to use CSV files, as they seem to have less trouble as Excel for some reason unknown to me. So, if you have an Excel file, I recommend first saving it as CSV. Also make sure the variable names are in the first row of the spreadsheet.

While there are other ways to import CSV files, I personally prefer the import wizard to writing code for this task.

- Click on File (upper left corner), and select Import Data, and you should see (after selecting CSV)



- Click on Next, and specify where you have saved the CSV file by clicking on Browse, and then click Next. Now you should see



- This is when you can choose your sta310 library, and then under Member, you can type the data set name you will use in SAS. Click Finish, and you have successfully imported the CSV file into SAS.

You can use the viewtable feature to see the data, but always remember to close it when done viewing. Double click on your library name, and you should see an icon with your new data set name. Double click on that icon, and you can view your data, which is especially helpful after trying to create new variables, to verify that you got the result you wanted.

A common point of confusion for students is that they don't realize that there is no reason to reimport the data if you put it in a permanent SAS library like sta310. Just refer to the data using

libraryname.datasetname.

2. Data step

I would first always recommend you include comments in your code, to make it easier for me to read, and for you to remember what the goal was. Comments can be included in SAS code by using an * at the start of the line. Examples are included below.

Data steps are useful to create new data sets from existing SAS data sets, where you might want to subset the data, or create new variables from existing variables. Suppose we already have a data set imported into SAS called example, and stored in the sta310 library, and we want to base the new data set called newdata based on the data set sta310.example. To create a new data set called newdata (you can call it whatever you like), use the following:

```
*creating new data set from sta310.example;  
data newdata;  
set sta310.example;  
  
run;
```

This code has now created a data set called newdata, and it is stored in the temporary SAS library called Work, and has all the variables from the sta310.example data set. If you want to avoid having to run this code again, you could always tell SAS to put your new data set into the sta310 library by using:

```
data sta310.newdata;  
set sta310.example;  
  
run;
```

Now to create new variables, you can just write the necessary code between the set and run statements. For example, if the data set sta310.example has a variable called inches, and you wanted to create a new variable called feet, use

```
*creating new variable feet from inches;  
data newdata;  
set sta310.example;  
feet=inches/12;  
run;
```

The data set called newdata will now have all the original variables from sta310.example, and also the new variable called feet.

As another example, if you wanted to create indicator (dummy) variables for the purposes of including a categorical variable called race ("black", "white", "other") into a regression model, which would require two indicator variables called black and white (where "other" is taken to be the reference category), that could be accomplished as

```
*creating indicator variables, using "other" as
  reference category;
data newdata;
set sta310.example;

if race="black" then black=1;
if race="white" or race="other" then black=0;

if race="white" then white=1;
if race="black" or race="other" then white=0;
run;
```

When creating a new variable from an existing categorical variable, the name of the category in quotes must match the exact spelling in the data set.

I discourage the use of the "else" statement, unless you have verified there are no missing data in the variable you are basing the new variable on (here, the race variable). A more efficient and equivalent way of creating indicator variables is

```
*creating indicator variables, using "other" as
  reference category, without if-then
  statements;
data newdata;
set sta310.example;
black=(race="black");
white=(race="white");
run;
```

Regardless of how you create new variables, you should first look at the raw data to see if SAS did what you really intended before doing anything else!!!!

3. SAS procedures (proc's)

Once you have the data set in the form that you need, you will likely want to run analyses on it. That could include basic numerical and graphical summaries, performing hypothesis tests, fitting models, etc. There are a large number of SAS proc's, and they all have their own syntax. From your STA 216 experience, I assume you are familiar with the following SAS proc's:

- univariate
- means
- freq
- reg
- glm

I will demonstrate how to use all the proc's that are new for this class as the semester progresses. However, for example, if you wanted basic numerical and graphical summaries of a quantitative variable called inches, you could use

```
*summarizing variable inches;  
proc univariate data=sta310.example plot;  
var inches;  
run;
```

Proc freq would be used to get numerical summaries of a categorical variable (counts of how many individuals in each category).