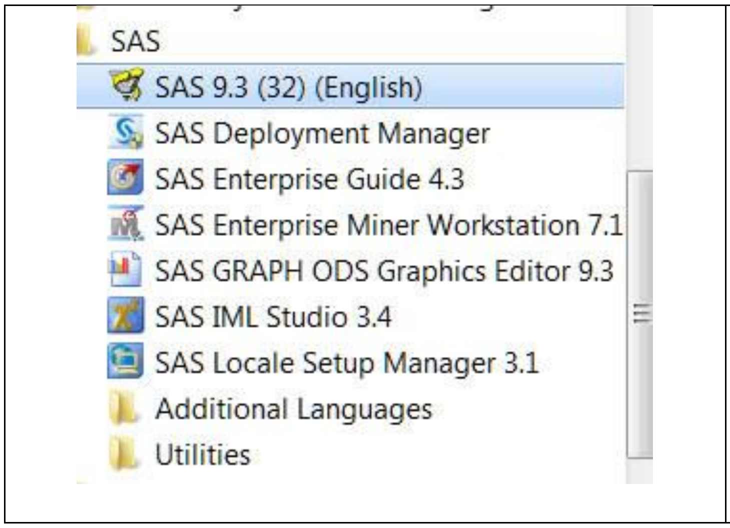
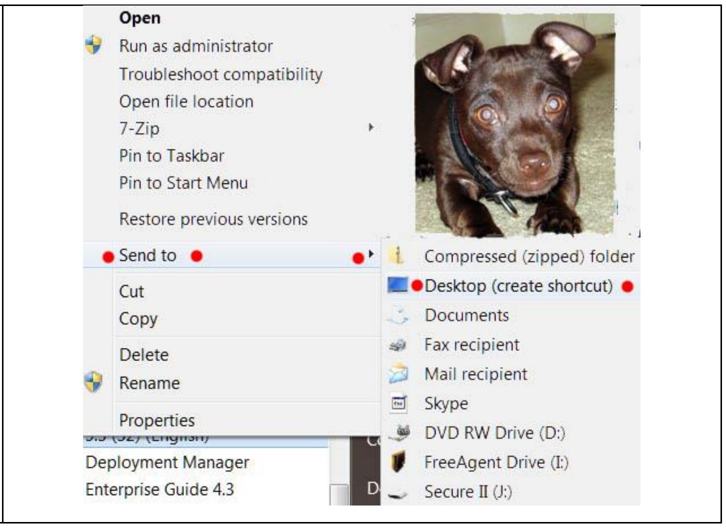
# An Introduction to SAS for Windows 9.3 and 9.4[©]

My first experience with SAS took place in 1977, at Miami University. In those days, we punched our programs onto Hollerith cards, with different patterns of holes coding different characters. Computing has evolved considerably since then. Most of my SAS experience has been on mainframe computers, but I shifted to the PC/Windows platform in the Autumn of 2000.

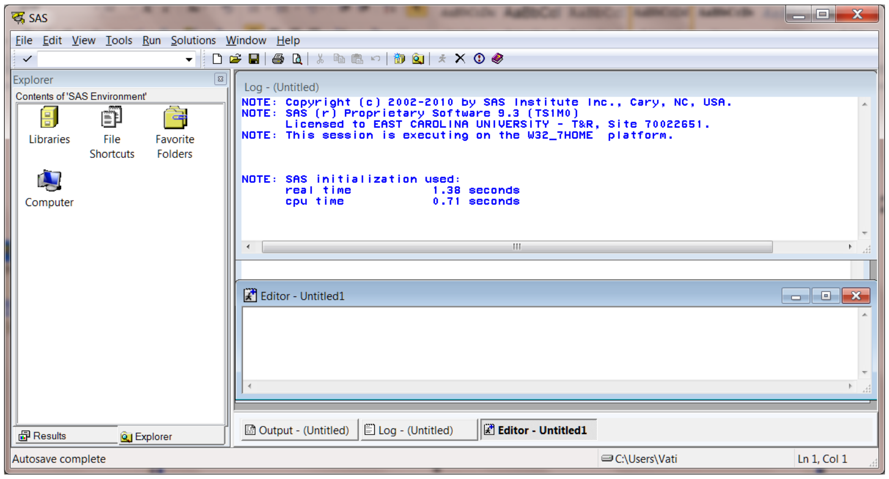
See this document about Directing New Output to a New Viewer File .

Let us start learning SAS by using SAS. Go to the start menu. Expand SAS. What you want is SAS 9.3 or 9.4. If you do not already have an icon for SAS 9.3 or 9.4 on your desktop, right click on SAS 9.3 or .4, click “Send to,” and then click “Desktop (create shortcut). SAS 9.3 will not run on Windows 10, but 9.4 will.





You can start SAS from the start menu or from the icon on the desktop. In our labs the exe for SAS 9.4 can be found in the Stats folder on the desktop. Here is what your window should look like now:



```
NOTE: Copyright (c) 2002-2010 by SAS Institute Inc., Cary, NC, USA.
NOTE: SAS (r) Proprietary Software 9.3 (TS1M0)
      Licensed to EAST CAROLINA UNIVERSITY - T&R, Site 70022651.
NOTE: This session is executing on the W32_7HOME  platform.


NOTE: SAS initialization used:
      real time           1.38 seconds
      cpu time            0.71 seconds
```

The enhanced editor is at the bottom of the screen, the log at the top, and the SAS Explorer and Results windows at the left. Click the X icon (in upper right corner of that window) to remove the Explorer and then the Results window. Now you have a better view of the editor and log windows. You can always get the Explorer and Results windows back via the View menu.

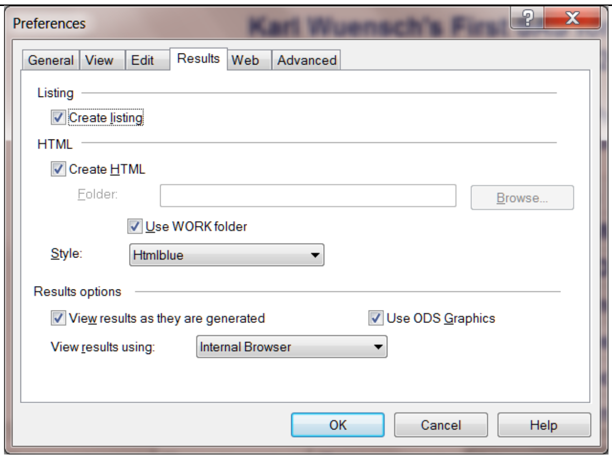
Let us use the editor. Place your cursor in the editor window and type the following commands/statements, ending each line with a stroke of the Enter key -- but replace my name with your name and my date with the current date, and note that what looks like a double quotation mark near the end of my last name is not, it is two single quotes, the way one tells SAS to print an apostrophe:

```
options formdlim='-' pageno=min nodate FORMCHAR="|----|+|---+=|-/\<>*";
Title 'Karl Wuensch''s First SAS for Windows Program';
 Title2 '27. August, 2010'; Run;
Data checking; Input ID $ Amount @@; Cards;
c407 5.00  e1 165.78  e2 6000.00  e3 35.13  c396 210.00  d1 2389.29  e4 1530.07
e5 36.38  d2 2389.29  e6 167.14  c409 70.88  pc1 62.53  d3 672.74  d4  2316.36
;
Proc Univariate plot;  Var amount; id ID; run;
```

The data are financial transactions from my checking account for six weeks in the year 2009. After you have entered the program (you could just copy and paste it from this document to the SAS editor), click on the running person icon which you should see on the toolbar. Your statistical output goes to the Results Viewer window, where you can view it.

In our labs, SAS is configured to produce HTML output, which can be found in the “Results” window after you run your program. If you want to save an HTML copy of the output, while in the Results Viewer, click File, Save As, select the desired output format, point to the desired folder, and then give it a file name (such as SAS-1). Alternatively, you can copy and paste the output from SAS to Word.

You can save them as HTML or as plain text. You can modify the default settings for the format of the output. By default, the format is HTML for code run from the editor and plain text (a “listing”) for code submitted batch. I have set SAS (on my computers) to produce both formats. Click Tools, Options, Preferences. After indicating your preferences, click OK. In the labs the setting is for HTML output only.



Listing
☑ Create listing
HTML
☑ Create HTML
Folder:
☑ Use WORK folder
Style: Htmlblue
Results options
☑ View results as they are generated
☑ Use ODS Graphics
View results using: Internal Browser

If you have enabled plain text output and want to save a copy of the plain text output, go to the “Output window,” click File, Save As, and then give it a file name (such as SAS-1). The default file type is lst, a listing, which is a plain text file, with hard carriage returns at the end of each line and spaces used for formatting. You can indicate into which folder you want the file saved. By default, it goes to the folder which was indicated when SAS was installed. From the output Save As window you can change the file type to RFT. This will produce an output file which can be opened by most any word processor – but will probably have to reduce the left and right margins so that the lines do not wrap. Also RFT files can be very much larger than docx files, so I discourage their use.

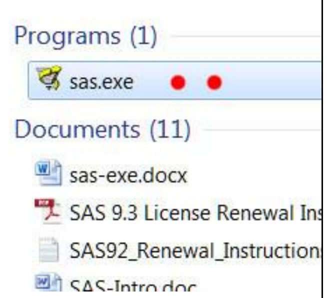
SAS puts the plain text output into the “Output” folder (if enabled) and the HTML output into the “Results” folder within SAS. Note that I refer to both plain text and HTML output as “output.”

When you open an SAS listing file with Word, open it as 'text only' the first time you open it with Word. Then select the entire document (CTRL-A) and change the font from whatever Word has selected (probably a courier font) to the font used by SAS, which is SAS MONOSPACE. If you do not make this change, you may find strange symbols in your document where there should be lines. You will probably also have to adjust the margins (I recommend .8 inch left and right), and the font size (I recommend 8 point), so that lines are not wrapped. Once in Word, you can do all sorts of editing: Annotate the output, delete unwanted items, put page breaks where you want them, use Word's drawing tools, insert objects with the equation editor or other Microsoft products, etc. Also, keep in mind that the SAS font will not display correctly when the Word document is opened on a computer where the SAS font has not been installed. Installation of the SAS font takes place during the installation of the SAS software. You can avoid this problem by saving your Word document as a pdf file. Also, you can add to the options line this code to produce a portable document (one formatted for proper reading even on computers without SAS monospace font):

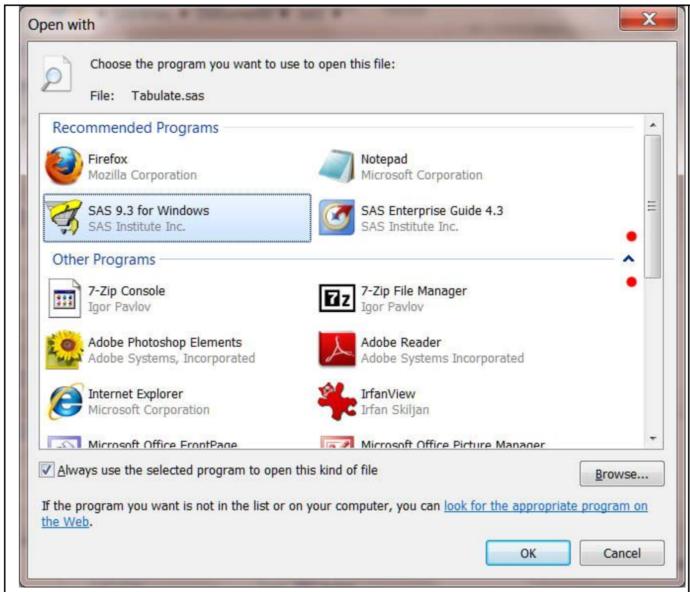
```
FORMCHAR="|----|+|---+=|-/\<>*".
```

You will probably also want to save a copy of your SAS program. From the Editor window in SAS, Click File, Save As, give it a name (SAS-1), and save it. It will go to the same default directory indicated above, unless you have pointed elsewhere. That program can be brought back into SAS simply by double clicking on its name in the Windows Explorer. It can also be brought into Word and edited there. You can also copy and paste back and forth between SAS and Word. You could, if you wanted to, write your SAS program in Word or any other text editor and then paste it into SAS.

**Batch mode.** When you have a SAS program in a file on your computer, double-clicking on the file name will submit the program to SAS in batch mode. SAS will run the code and then deposit the listing and the log in the same folder where the program file was. If SAS does not operate like this, but you want it to operate like this, you may need to change the program that is executed when you double-click on a file name. The first thing you need to do is locate “sas.exe.” From the Start Menu search box, search for SAS.exe. It should appear in the search results under “Programs.” Right-click on it and select “Open file location.”



On my home PC, SAS 9.3 was located in C:\Program Files\SASHome\x86\SASFoundation\9.3. The easiest way to associate SAS program files with sas.exe is the right-click on the name of a SAS program file, select “Open with” and then “Choose default program.” Click on the arrowhead to expand “Other programs.” If SAS is not displayed, browse your way to SAS.exe, and select it. If you do not do this, double-clicking on a SAS program file may bring it into SAS Enterprise Guide instead of sending it to regular SAS. I do not use the Enterprise Guide.



Open with

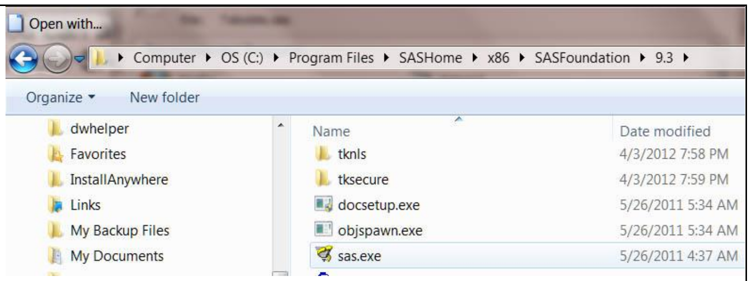
Choose the program you want to use to open this file:

File: Tabulate.sas

Recommended Programs

[✓] Always use the selected program to open this kind of file

If the program you want is not in the list or on your computer, you can look for the appropriate program on the Web.



Open with...

Computer ▸ OS (C:) ▸ Program Files ▸ SASHome ▸ x86 ▸ SASFoundation ▸ 9.3 ▸

| Name | Date modified |
|---|---|
| tknls | 4/3/2012 7:58 PM |
| tksecure | 4/3/2012 7:59 PM |
| docsetup.exe | 5/26/2011 5:34 AM |
| objspawn.exe | 5/26/2011 5:34 AM |
| sas.exe | 5/26/2011 4:37 AM |

After running your SAS program, take a look at the contents of the LOG window. Here are your program statements and notes (and maybe error messages) about them. It is easy to ignore the log, but that can be a big mistake. Just getting output does not mean that all went well. In the log from the program you just ran, there is a line which says:

"NOTE: SAS went to a new line when INPUT statement reached past the end of a line."

Often this would indicate a serious problem, but, as you will soon learn, for our current program it is not a problem. The log can be printed or saved as a text file also. If you are having problems with a SAS program and want to get help from me or another consultant, it is critical that you present your consultant with a copy of your SASLOG. The consultant may also need access to your data set and your program file. In our current program, the data set is inside the program file, but that is often not the case.

Now, let me explain the SAS statements in our program. The first line specifies three **Options**. **pageno=min** sets the first output page to 1. If you have already submitted other programs during your current session and you do not include this option, then your output for the current program will start with a page number one greater than the page number of the last page of the last submitted program's output. **nodate** simply prevents SAS from putting the current date on every page of output. **formdlim='-'** replaces page breaks with a dashed line, which can save you a lot of paper, and is highly recommended. If a figure gets spilt across two physical pages, you can always edit the file in Word to fix that. Notice that the options command ends with a **semicolon**. This punctuation is very important. Every SAS command must end with a semicolon.

The **Title** statement indicates, within single quotes, a title to be printed on the first line of every page of output. When you are programming for my classes, I expect your name to be in such a title statement. The title text is enclosed in single quotes. The **Title2** statement gives text to be printed on the second line of each page of output. I have found that a **run** statement after title statements makes it more likely that they will appear where you want them to appear.

“**Data checking;”** tells SAS to start the data step, the part of the program that defines the data. "data" is the command word, "checking" is the name I close for this set of data, since the scores are amounts of financial transactions from my most recent checking statement. The **Input** statement tells SAS that for each transaction there are two variables, id and amount. It is highly recommended that you **always include an ID number for each observation in your research data**. One of the first things you should do after entering research data into the computer is screen them for values that suggest errors. When you find such suspicious values, you can use the ID number to go back to the source of the data and see if the suspicious value is correct. The **$** following “ID” tells SAS that the ID variable is alphanumeric, that is, may contain letters, not just numbers.

The @@ tells SAS that there is data for more than one transaction on each line in the data stream. Without the @@, SAS would, after reading two data values on a line, skip to the next line. We used the @@ so we could pack several pairs of data on each line. This is what resulted in the Note in the SAS log which I mentioned earlier. Note that we have three commands on this line, each ending with a semicolon. The last command, **Cards**, (referring to the Hollerith cards I mentioned earlier) tells SAS that the following lines contain data, not program statements.

Note that after the last line of data there is **a line with nothing but a semicolon**. Although not usually necessary, this is good practice, as it tells SAS that you are done feeding it data, and that subsequent lines will once again be program statements. As soon as SAS finds a semicolon, it returns to command mode -- but in some programs the first statement after the data might be so long that it takes two lines, in which case the first line would not have a semicolon, and SAS would mistake it for data if you had not used the "semicolon alone" technique I showed you here.

**Proc univariate plot; Var Amount; id ID;** asks SAS to compute a variety of descriptive and inferential statistics and plots on the variable 'amount', using the variable 'id' as an identifier variable. The **Run** statement at the end tells SAS to go ahead and run the procedures we have indicated.

Let us now look at the output. Under moments, note that the data are somewhat skewed, positively. SAS computes the **coefficient of variation** as 100 times the ratio of the standard deviation to the mean. The **uncorrected SS** is $\sum Y^2$, while the **corrected SS** is $\sum(Y-\mu_Y)^2$. The large value of **kurtosis** (the $g_2$ statistic) makes me suspect outliers, which the large, positive value of skewness (the $g_1$ statistic) makes me suspect the outlier(s) are in the upper tail. The stem and leaf plot (or horizontal histogram) and the box and whiskers plot confirm these suspicions. Under tests for location, **t** tests the null hypothesis that the distribution from which the data were randomly sampled has a mean of zero. The **sign** test tests the hypothesis that half of the scores in that population are positive, half negative. The **signed rank** tests the hypothesis that the population scores are symmetrically distributed about zero.

Now, let us talk programming again. We could have arranged our data more simply, with one line for each transaction, and only two scores on each line. In that case we would not have put the @@ in the input statement, and our data lines would look like this:

```
c407 5.00
e1 165.78
etc.
```

Note that we have left at least one blank space between each data value and the next data value. This is called **list input**. If you have missing data for an observation and are using list input, you must represent each **missing value** with a **period** in the data stream.

With large data files, **column input** (also known as fixed field input) is a better choice. With column input, each variable has a field of certain columns in which its values must be. For example, suppose that we decided to enter my financial transaction data this way:

| Col | Variable | Type |
|---|---|---|
| 1-4 | ID | alphanumeric |
| 5 | blank | |

| | | |
|---|---|---|
| 6-12 | amount | numeric |
| 13 | type | alphanumeric |

The code for type of transaction will be:

D = deposit or other credit
I = investment
C = credit card payment
X = other expense  (did not use letter O, as it is too easily mistaken as number 0)

Our data file is now going to look like this, in courier font to make the column formatting clear.

```
c407    5.00X
e1    165.78X
e2   6000.00I
e3     35.13C
c396  210.00X
d1   2389.29D
e4   1530.07C
e5     36.38C
d2   2389.29D
e6    167.14X
c409   70.77X
pc1    62.53X
d3    672.74D
d4   2316.36D
```

and our input statement will look like this

**Input ID $ 1-4    Amount 6-12    Type $ 13;**

While I could put these data in the program file, after a cards statement, like we did earlier, I have instead put it in a separate data file. I created that data file in Word, and when I saved it I told it to save the file with the name "SAS-2.dat" and as type **Text Only**.

**Preparing the computer to work with .dat files.** Files typically have an extension at the end of their name, following a dot. This extension is used by many programs to indicate what type the file is (data, SAS program, Word document, etc.). It is helpful to be able to see the extension so that you know if a pesky Microsoft product has added an extension to a file name and not bothered telling you about that. If it has, and you don't know about it, you may refer to the file by a wrong name when writing an INFILE statement. Sometimes Windows will even treat an existing dot with extension as part of the file name and add an extension to that. To be sure that you can see what the extension is, **tell your computer not to hide the file extensions**. If you are using a computer in a lab, you may find that the file extensions have been hidden and you cannot change that setting. This may create problems for you -- you will just have to be mindful that a file that appears in a folder with the name DUH and with an entry of DAT in the Type column really has a name of DUH.DAT. In an INFILE statement you would need to identify the file as DUH.DAT, not as DUH. If the file were to appear in the directory as DUH.DAT with extensions hidden and TYPE shown as DAT, the real name of the file would be DUH.DAT.DAT, and that is how you have to refer to it in your INFILE statement.

**Viewing dat files.** Your operating system, having assumed that you are an idiot, may be reluctant to allow you to open a dat file with a the Notepad or other text editor. One way to avoid this is to change the extension from dat to txt. Of course, when you do that the operating system will tell you “hey, moron, remember that changing a file extension can make the file screw up.”

**Downloading the .dat file.** I have uploaded the data file, SAS-2.dat, and the program, SAS-2.sas, to the core server. Go to http://core.ecu.edu/psyc/wuenschk/StatData/StatData.htm and get the data -- download them to your hard drive or to your flash drive. Once the StatData page has opened in your browser, point at the file you wish to download and right click the mouse. You will get a drop down menu. From that menu select "Save Target As" (IE) or “Save Link As” (Firefox” and then point at the drive/folder where you wish to store the data file. It might be a good idea to browse the folder to which you directed the download, just to see that the file arrived there OK and that a computer gremlin did not do something nasty like append a “.txt” file extension, change a hyphen to an underscore, or other mischievous changes that could cause you problems later when you try to point SAS to the data file. If the name has been corrupted during the download, just change it back to what it is supposed to be (right click on the file name, select “rename,” change the name, hit the enter key).

**Downloading the .sas file.** After getting the .dat file, point your browser to http://core.ecu.edu/psyc/wuenschk/SAS/SAS-Programs.htm to get the SAS-2 program. You can download the program to disk or you may be able to bring it into SAS directly just by telling your browser to open it (depending on how your antiviral software is set up and certain operating system and SAS settings on your computer). If you saved the data in a location identical to that specified in the INFILE statement of the program, then you can just click the running man and the program runs. If not, you first must edit the program (you can do that from within SAS' editor, or use another editor, such as Word or Notepad, if you prefer) and then run it. For example, if you saved the data file on a flash drive identified by the system as the F drive, you should change the infile statement to **Infile 'F:\SAS-2.dat';**

**Running SAS programs from the server.** You may or may not be able to run SAS programs directly from the server. That is, you may be able to go to my SAS programs page on the web, Microsoft Internet Explorer, click on the link to one of the SAS programs, tell it to "Open this file from its current location," and then the program file is opened in SAS' editor – or submitted in batch mode.

After running the program, look at the output and the program while reading my explanation. In the options statement I asked to have the **date** printed on each page. The **infile** statement points to the external data file, telling SAS that the data file is named SAS-2.dat, and it is in the directory StatData on my C drive. I have already explained the input statement. The **Label** statement gives a label to the variable 'Type'. **Proc Means** gives us some basic descriptive statistics about the amount variable. There are other optional statistics (including skewness and kurtosis) available with proc means. **Proc Sort** instructs SAS to sort the data according to the value of the 'type' variable. This is done prior to using the **By** variable again with Proc Means. The second invocation of Proc Means, using "By Type," yields basic descriptive statistics for the 'Amount' variable at each level of the 'Type' variable.

You might wonder why I prefer SAS over SPSS, in a department where most of the faculty use SPSS. One reason is that it is the first powerful statistical tool I learned. A second reason is that it is still syntax based (you type commands and statements rather than point and click). I think there is danger in being able to point and click one's way to a statistical analysis for which one cannot even write a model statement -- without knowing the underlying model, how can one hope to understand statistical output from an analysis based on a mathematical model. As you will see, the statements we type to obtain statistical analyses in SAS are often statements of the structural model we are testing. A third reason to learn SAS instead of, or in addition to, SPSS is that SAS is very commonly used in the nonacademic world, but SPSS is not. The student who enters the workplace with knowledge of SAS has a distinct advantage over those who know only SPSS. In fact, a number of our graduates have gotten good employment based not on what psychology they learned but rather on their skills in SAS. In fact, they often have starting salaries that exceed those of their professors. Dale Glaser wrote an article (in the November/December 2001 issue of *Psychological Science Agenda*, which is an APA publication) about statistical and methodological consulting as a career for those trained in psychology. He advised: "virtually all the large manufacturing/pharmaceutical concerns require SAS programming experience. Thus, the more software you can learn (besides SPSS), the more marketable you will become." Also, if you think you can avoid syntax by using SPSS, think again. There are lots of things in SPSS that you cannot do by pointing and clicking but can do with syntax.

**Importing Data to SAS**

It is easy to bring into SAS data from SPSS, Excel, and many other formats, and equally easy to export from SAS to other formats. See my documents for details.

- Return to Karl's SAS Lessons Page
- Directing New Output to a New Viewer File – read this doc if you are having problems clearing the viewer.
- More on the difference between Listings and Results
- Stem and Leaf Plots