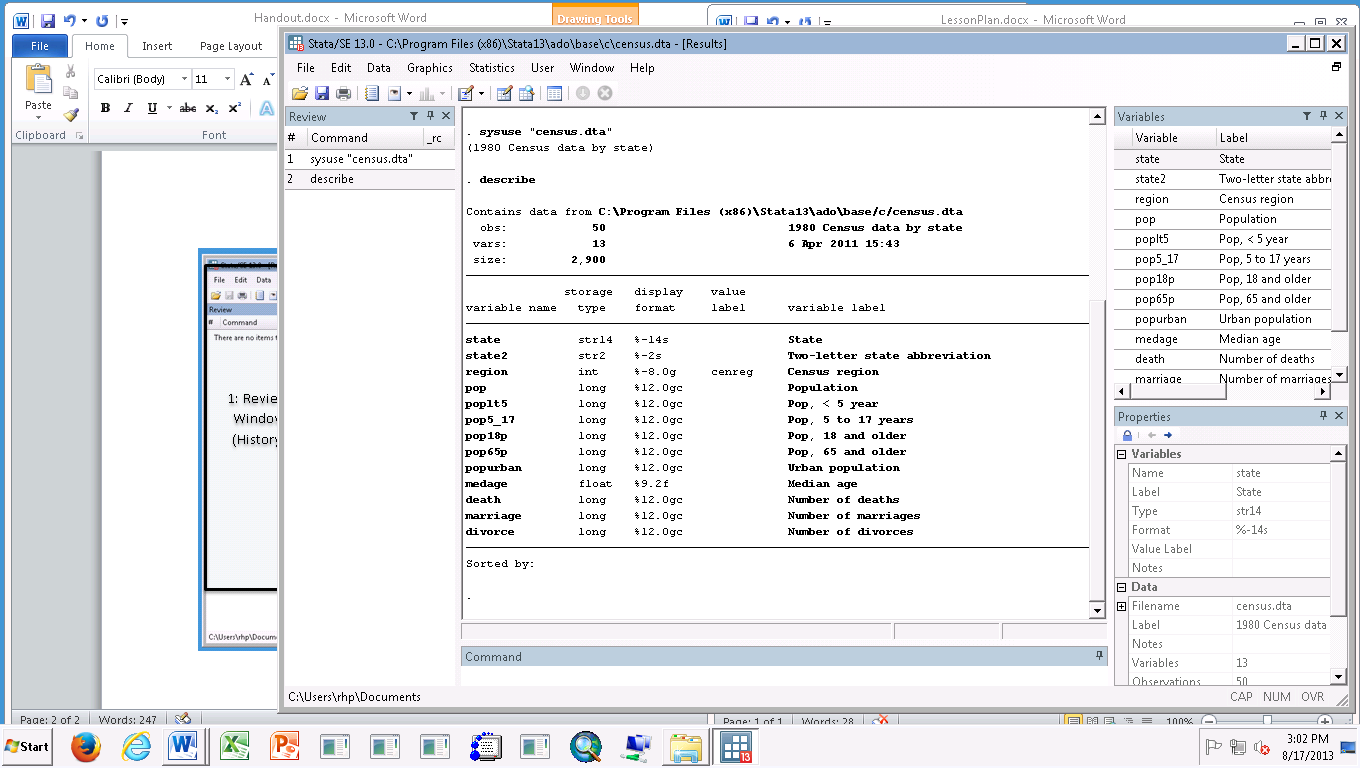
## Introduction to Stata: Stata Basics

## The Stata Interface

*Begin by opening the Stata program.* By default you see five empty windows, which will display information about the data and commands for the current Stata session.



**5: Properties**

**4: Variables List**

**3: Review  
Window   
(History)**

**2: Results Window (Output)**

**1: Command Line Window**

**1: Command Line Window** – type commands here

**2: Results Window** – echoes typed commands and prints their output

**3: Review Window** – saves a list of the commands you typed; **red** commands produced errors

**4: Variables List**  - list of the loaded variables and their labels

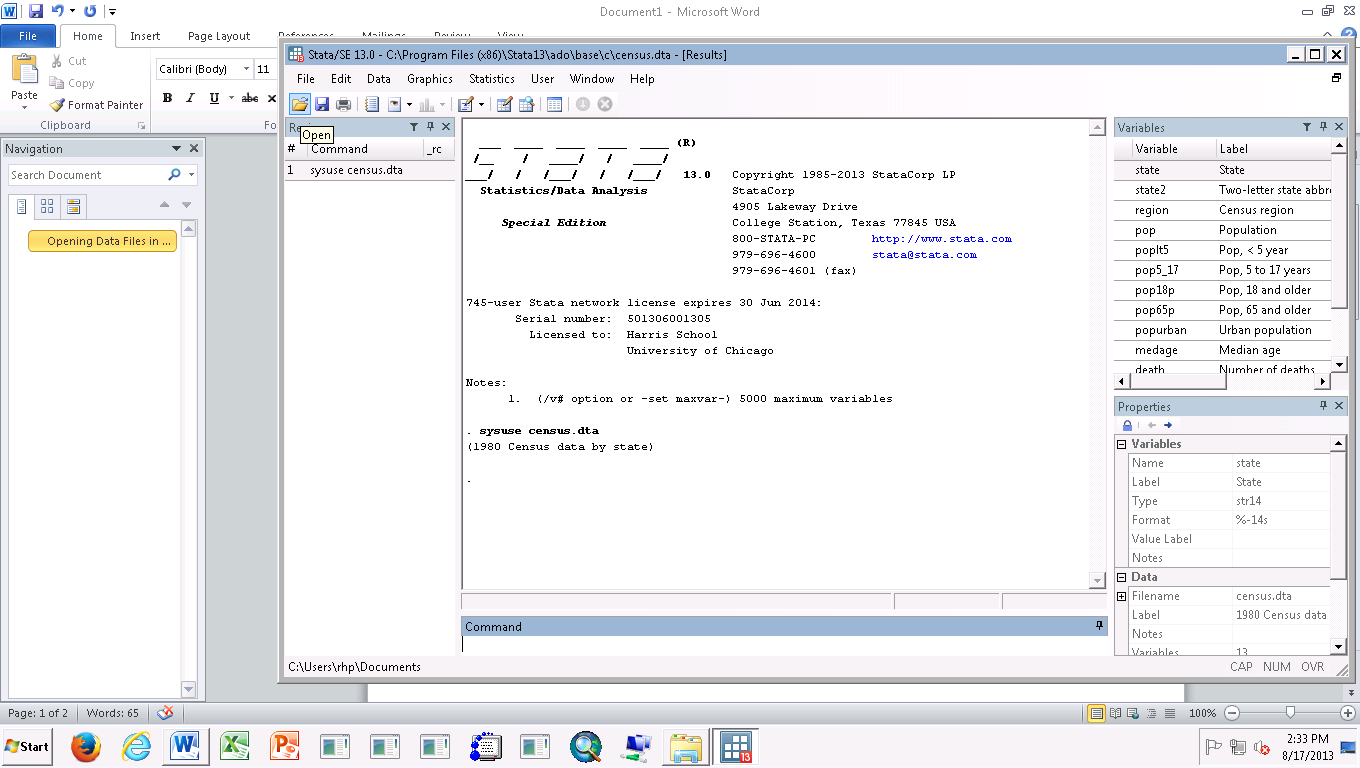
**5: Properties** – additional details about each variable and about the data file

From this point, you can either open an existing data set if you want to work with real data or you can generate an example data set to explore statistical behavior of simulated variables. This training will expose you to both. The second training will focus on simulated variables, while the third will provide a deeper background on working with real data.

## Opening Data Files

Stata data files use the “.dta” file extension.

Any of these approaches will open an existing Stata data file:

* From an open Stata session, use the menus to select File -> Open
* Click on the folder icon at the top left of the Stata session
* Double click on the file itself. A new Stata session will open.
* Type: **use “<path>\<filename>.dta”** into the command line window. For example, I have the file saved in “U:\StataTraining”, so I type: use “U:\StataTraining\census.dta”

Stata also has a number of built-in datasets and online datasets that can easily be loaded (the data is usually fictional). If you want to practice with Stata, consider using one of these. Later, you’ll see that all of the help files have example commands you can execute using one of these datasets. To load one of the example datasets, type **sysuse** **<filename>** (built-in) or **webuse <filename>** (online). Type **sysuse dir** for a list of the built-in datasets or visit <http://www.stata-press.com/data/index.html> for the online ones (typically, you’d be interested in the Stata 13 -> Base Reference Manual option).

For this example, we’ll use one of the build-in datasets, which is data about cars reported in Consumer Reports in 1978. *In the command line window, type*:

**sysuse auto**

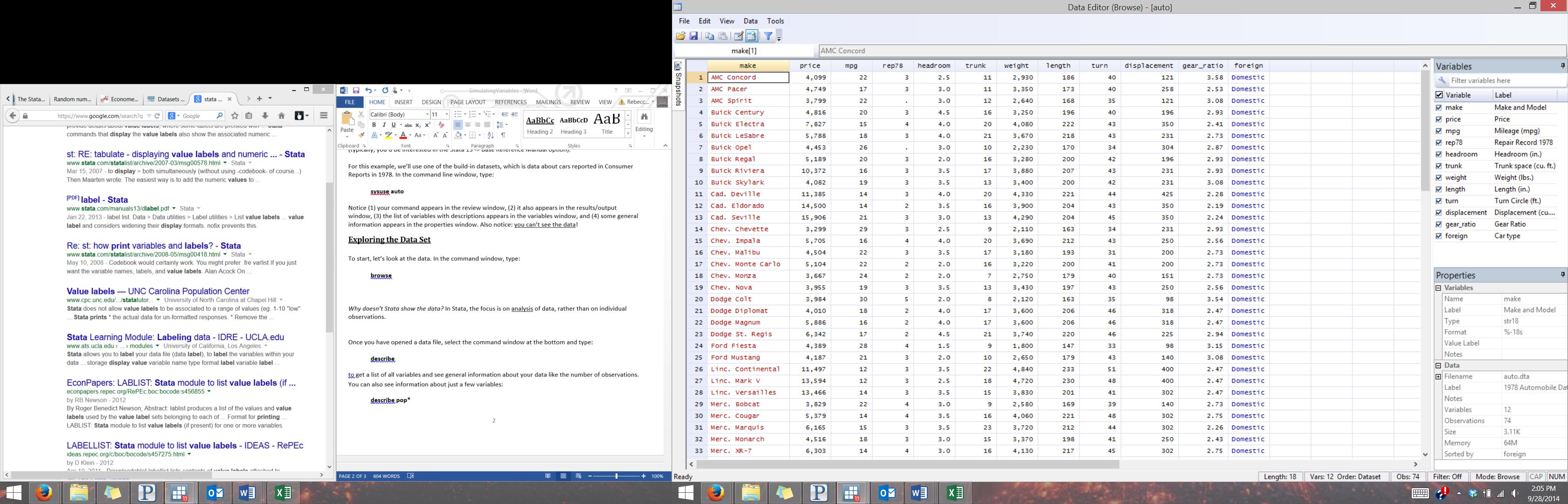
Notice (1) your command appears in the review window, (2) it also appears in the results/output window, (3) the list of variables with descriptions appears in the variables window, and (4) some general information appears in the properties window. Also notice: you can’t see the data!

## Viewing the Data Set

To start, let’s look at the data. *In the command window, type*:

**browse**

A new window will appear, where each row is one type of car, and each column contains a variable and its value for each of the cars. Notice the variable names are listed along the top and the rows are numbered. We’ll talk about each row as an “observation” and the numbers as the “observation number.”



You can also get the data browser by choosing Data->Data Editor->Data Editor (Browse). This produces a data window more like what you might see in Excel, with the additional Variables and Properties Windows on the right.

All Stata datasets will have this format. If you are used to working with data in Excel or another spreadsheet program, it might help to think of this as the “behind-the-scenes” spreadsheet. Consider these examples, which you can see by loading the relevant example dataset:

* For data about individual people, you would have a row for each person and columns for things like their age, race, marriage status, etc. (**sysuse nlsw88**)
* For data is about the life expectancy in various countries, each row would be for a country and you would have columns for the country name, its region, the life expectancy and other values of interest like the rate of population growth (**sysuse lifeexp**)
* For observations of the surface temperature of the ocean, each row would be for a particular location, with columns for the latitude, longitude, observed temperature, etc. (**sysuse surface**)
* For a longitudinal survey, which re-interviews the same people at various points in time, each row would be for an *observation*, so the same person would have multiple rows, one for each point in time. **webuse psidextract** will show various data from the Panel Study of Income Dynamics survey, where the id variable identifies the individual and the variable t identifies the point in time when the interview was conducted, with the remainder of the columns containing variables like wage, occupation, years of experience, etc.

*Now, CLOSE the browse window.* You will almost never use it. It’s not an accident that Stata does not display the data by default. Why? In Stata, the focus is on analysis of data, rather than on individual observations. You generally don’t care about the value of a variable for a particular observation (e.g. the miles per gallon for the Buick Skylark) – you’re usually interested in summary values (e.g. the average miles per gallon for all cars).

## Exploring the Data Set

When you open a new dataset, you’ll want to spend a little time exploring the data. For larger datasets, the **browse** command is a BAD idea – it’s hard to get a good sense of anything if there are hundreds of variables and thousands of observations.

*You should still be in the automobile data set. If you loaded any of the other examples, you’ll want to re-load the auto data set by typing* ***sysuse auto*** *again.*

To get a list of all variables and see general information about your data*, select the command window at the bottom and type:*

**describe**

You can also see information about just a few variables *by typing:*

**describe price**

**describe price mpg length**

Notice that it gives the type of variable – string, integer, decimal (called float), binary (called byte) – as well as information about how it will display the variable, any relevant value labels (more on this later), and a description of the variable.

To get an idea of what your data is like, you can also *print out the first five observations:*

**list in 1/5**

If you have a particular observation number you want to see, *try*:

**list in 25**

If you want to see observations with gas mileage greater than 34 mpg, *try:*

**list if mpg>34**

If you only want to see a few variables for the full dataset, *try:*

**list make mpg weight**

To see the same variables for a subset of the dataset, *you can do*:

**list make mpg weight in 1/5**

**list make mpg weight in 25**

**list make mpg weight if mpg>34**

(Don’t worry if the syntax for the last few commands isn’t clear – more on it in the third training.)

## Creating Variables

To create a new variable, you will use the **generate** command. You need to tell Stata the variable’s name and how to calculate it. For example, you might want to know the ratio of a vehicle’s weight to its gas mileage. *Type:*

**gen wt\_ratio=weight/mpg**

Scroll to the bottom of the Variables window; you will see your new variable. Notice:

* We can type **gen** instead of **generate**. Stata commands are often shortened to help you work faster. For example, you can use **li** instead of **list**, **d** instead of **describe**, and **br** instead of **browse**. To find the abbreviation for a command, look in the **help** file to see which letters are underlined in the command.
* Stata does not have autocorrect, and it’s case sensitive. If you type Gen instead of **gen** or wieght instead of **weight** you’ll get an error message! For more complex commands, you’ll often get an error message on your first try, and need to go back and double check what you typed.
* We picked the new variable’s name (wt\_ratio). It could be whatever we wanted, within reason (no duplicate variable names, no spaces, no symbols, can’t start with a number, etc.)
* The math worked intuitively: for each observation, the value in wt\_ratio is calculated as the quotient of the values of weight and mpg *for that observation*.

Math operations use the symbols you might expect, and you can use parentheses to enforce the order of operations you want:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Addition | + |  | Subtraction | - |  | Exponents | ^ |
| Multiplication | \* |  | Division | / |  |  |  |

***Exercise****:* The price variable in the data set gives the purchase price in 1978 dollars. To convert these values to 2013 dollars, you would multiply by 3.573 (the ratio of the 2013 and 1978 Consumer Price Index). On your own, create a variable containing the inflation-adjusted price. Name it price\_13.

*Check your work:* The new variable (price\_13) should have a mean of $22,028.

***Extensions:*** Look at the “Stata Commands” handout. Explore using rename, label var, replace, and egen.

## Saving Your Work

*To save your data file, choose a convenient location to save the data on your computer, and find the path. The example shown here is for saving the data to your “U:” drive on the Harris computers in a folder you could create called StataTraining. Now type:*

**save “<path>\auto\_edited.dta”**

e.g.: **save “U:\StataTraining\auto\_edited.dta”**

You can also use the menu options (file->save) or type control-s. If you’re not at Harris, choose an appropriate location to save your Stata files.

## Getting Help

Stata has extensive documentation about its commands. If you know the name of a command you want to learn more about, type **help <command name>** to see the help file. If you don’t know the name, you can still get assistance on a topic by typing **search <topic>**. For example, if I want to know more about graphs, I might type **help graph** or **search graph**.

There are also extensive online resources providing more accessible help with Stata – a few useful ones are linked from the Chalk page for this training.

***Exercise 1:*** Download and explore another of Stata’s built-in data sets, called nlsw88. Get a list of all the variables with descriptions. Print out the value of the variables for the first observation in the data set.

## Introduction to Stata: Data Analysis

For basic data analysis, you’ll want to know a few commands. First, load the built-in census.dta file:

**sysuse census**

Note that some of the later commands rely on variables created in the generate examples.

|  |  |  |
| --- | --- | --- |
| **generate** | Create a new variable. The new variable can be set equal to a constant, another variable, and mathematical expression, a function, etc. The () creates a 0/1 indicator variable, set to 1 when the statement is true.  The last creates a random variable with a mean of zero, standard deviation of 1. | **gen constant=5**  **gen ln\_pop=ln(pop)**  **gen death\_rate=death/pop\*100**  **gen high\_death=(death\_rate>1)**  **gen share\_older=pop65p/pop\*100**  **gen random\_var=invnorm(uniform())** |
| **summarize** | Print summary statistics for variables. Alone, it prints summary statistics for all variables. | **sum**  **sum pop5\_17**  **sum pop5\_17, detail**  **sum pop\***  **sum pop death marriage divorce**  **bys region: sum pop** |
| **tabulate** | Print the number of observations in each category | **tab region**  **tab region urban**  **tab mathprob lowinc\_cat, chi all** |
| **table** | Print a table of selected summary statistics; defaults to frequencies, use the contents(<stat><var>) option to specify other statistics | **table region**  **table region, c(n pop)**  **table region, c(m pop p10 pop med pop p90 pop)**  **table region, c(m death m marriage m divorce)** |
| **tabstat** | Print specific summary statistic(s). Options let you get the statistics for chosen subgroups. | **tabstat pop, by(region) stat(med)**  **tabstat pop, stat(mean, med, p10, p90)**  **tabstat pop death marriage, by(region) stat(mean, med, p10, p90)** |
| **histogram** | Plot the density or frequency distribution for a variable. | **hist pop**  **hist death\_rate, normal bin(20)**  **hist region, discrete frequency**  **hist pop, by(region)**  **hist death\_rate, normal scheme(s2mono) title(Distribution of death rate) ytitle(Density) xtitle(% of pop that died)** |
| **scatter** | Plot two variables against one another | s**catter share\_older death\_rate** |
| **graph & twoway** | Both are powerful commands with many options for plotting two variables against one another. | **graph bar share\_older death\_rate, by(region)**  **twoway (scatter math3\_fail t\_salary) (lfit math3\_fail t\_salary)** |

## Understanding Command Structure

Suppose you wanted to know the median population by region. There are at least four easy ways to do this:

**sum pop if region==1, d**

**bys region: sum pop, d**

**tabstat pop, by(region) stat(med)**

**table region, c(med pop)**

The first is using symbolic logic, telling Stata to give you the detailed summary for *only* observations in region 1. You would have to repeat this command four times: once for each region.

The second uses the “bys” option to do exactly that – Stata subsets the data into groups for each category of the region variable, and then does the command on each of those subsets. IMPORTANT: always use “bys” rather than “by”. The “by” option assumes you’ve already sorted the data into subsets.

The third is a separate, very powerful command, which lets you get a variety of statistics across a categorical variable. The by(variable) option tells it which variable to use for subgrouping, and stat(med) specifies which statistics you want. To get a sense of everything **tabstat** can do, type:

**help tabstat**

The fourth command allows you to do similar things, with slightly different syntax. Its help file is similarly illuminating.

***Exercise 2:*** Again use nlsw88. Find the mean for all the variables. For which variables is the mean a useful number? Find the median age and hours of work for the full sample. Find the median age and educational attainment for each of the race categories. Produce a plot that will illuminate the relationship between hourly wage and years you’ve held your job.

***Exercise 3:*** Try each of the approaches above to find the mean educational attainment(grade) for states in the South compared with those not in the South. Look in the tabstat help file to learn how to specify that you want the mean.

***Extensions:*** Try this command: **tabstat age grade wage hours, by(south) stat(mean, med, p10, p90)**. Notice you can use it for many variables and many statistics at once. Try some of the example syntax you see in the **tabstat** help file.

## Do Files and Log Files

You can cut & paste output from Stata into your homework. DON’T!

* You can spend a long time figuring out how to do something, and then forget it before the next homework. You want to be able to go back and see what you did.
* If you misread a value from the Stata output and then you’re checking your solutions with your classmates, it’s hard to know who has it wrong. You want to be able to go back and see the full output after your Stata session is over.
* If you realize you did something wrong, you don’t want to have to go back and re-create your work. You want an easy way to re-do everything quickly.

Stata solves these problems for you:

Do File – a text file listing the commands to run, including comments about what they do.

Log File – a text file of the output that appears in the Results Window

To get an idea of how a .do file works, you can look at the file I used to create an urban categorical variable for the census data. To see what that file actually did, type:

**doedit “urbanicity.do”**

Spend some time looking at the file. Notice:

* Look for some new commands (for example, find the **cd** command and read the comment).
* My comments are in green – anything in green will not be input into the command line.
  + // comments the rest of the line
  + Starting a line with \* comments that line
  + /\*Write a long comment by enclosing it \*/
* Special words that Stata knows are in blue, including command names and “if”.
* Strings (things between “quotes”) are in red.
* Everything else Stata will put into the command line is in black.
* The do file automatically opens a new log file and closes it at the end of the file:

**log using “urbanicity.log”**

... etc. …

**log close**

**ALWAYS, ALWAYS, ALWAYS work in a .do file, not in the command line!**

**ALWAYS, ALWAYS, ALWAYS log the output from your sessions!**

Professors often require you turn in your do files, and having them makes life much, much easier.

**Exercise Solutions**

***Exercise 1:***

sysuse nlsw88  
 describe  
 li in 1

***Exercise 2:*** Mean isn’t helpful for idcode, race, industry or occupation because the numerical values are not cardinal (nor even ordinal). While married, never\_married, collgrad, south, smsa, c\_city, and union are not cardinal, they are binary (only take the values 0 or 1), so the mean is the percent of the sample with the value 1 for the variable.

sysuse nlsw88  
 sum  
 sum age hours, d  
 table race, c(med age med grade) //there are multiple correct ways to do this  
 scatter wage tenure

***Exercise 3:***

sum grade if south==0 and sum grade if south==1  
 bys south: sum grade  
 tabstat grade, by(south) stat(mean)  
 table south, c(mean grade)