Introduction to Computing for Sociologists
Neustadtl

Examining Data

With a new dataset in hand your first task is to understand what the data actually represent—how concepts are operationalized into variables. The following briefly describes the ways to begin understanding your data. You can use the interactive help system within Stata to read about the details of each data entry method. For example, to learn more about the Stata command `list` you would enter "`help list`" in the command window. New in version 11 is the complete set of documentation in PDF format. To look at this documentation follow this path on the Stata toolbar: Help►PDF Documentation.

`describe` (`help describe`)

- `describe` produces a summary of the dataset in memory or of the data stored in a Stata-format dataset
- The following information is provided: variable name, storage type, display format, value label, variable label

`list` (`help list`)

- `list` displays the values of variables
- If no *`varlist`* is specified, the values of all the variables are displayed
- There are lots of options—the *`if`* and *`in`* options are useful

`summarize` (`help summarize`)

- `summarize` calculates and displays a variety of univariate summary statistics
- If no *`varlist`* is specified, summary statistics are calculated for all the variables in the dataset

`tabstat` (`help tabstat`)

- `tabstat` displays summary statistics for a series of numeric variables in one table, possibly broken down on (conditioned by) another variable. This is a more powerful alternative to `summarize`

`tabulate` (`help tabulate oneway`)

- `tabulate` produces one-way tables of frequency counts
- Many inferential statistics (e.g. chi-square, gamma, etc.) are available for twoway tables

`codebook` (`help codebook`)

- `codebook` examines the variable names, labels, and data to produce a codebook describing the dataset

Creating the Dataset

The following program will read in a subset of the General Social Survey:

```
local gss "http://terpconnect.umd.edu/~smilex3/GSS-Cumulative-72-12.dta"

#delimit ;
use year
    cohort
    socbar
    socrel
    socommun
    socfrend
    race
    age
    sex
    educ
    attend
    marital
    childs
    premarsx
    reliten
    childs
    polviews
    sexfreq if race!=3 using "`gss'", clear;
#delimit cr
```

Note the use of the `delimit` command to make the program easy to read. The *`if`* and *`using`* options drop respondents coded 3 on the variable race and points Stata to the location and name of the file containing the GSS data, respectively. Use `help operator` to see what "`!=`" does (i.e. not equal to). See the help file for `delimit` and `use` for more information. Finally, this program uses a *local macro* to assign the data file location and name to the macro `gss`. This will be briefly reviewed in class and covered more thoroughly later.

Create a Stata program file called "SOCY699C-CREATE.do" that reads in these variables. You will use these data for the remainder of this assignment.

describe

The `describe` command is usually the place to begin examining a new dataset. Using this command you can learn whether a variable is numeric or string. For example, consider the representation of gender. Assuming we can differentiate male and female is the variable “gender” coded as numbers (1=Male, 2=Female) or as strings (“Male” and “Female”)? Typically, strings cannot be used for statistical analyses.

Use the `describe` command and:

1. Attempt to define what the variables *age*, *sex*, *race*, and *sexfreq* measure. Your answer will probably be imperfect, but base your answer on the information provided by `describe`.
2. Describe the difference in storage types for the variables year and sex. See `help data_types` for more information. What can you infer from this difference?

list

The `list` command is simple but can produce enormous, difficult to read output. For example, entering list in the command window without any options will display values for every variable in the dataset currently in memory. With a dataset that has thousands of variables and tens of thousands of cases this is a mess. If you get a seemingly endless display of data scrolling across your screen you can stop the command from completing by entering CTRL-BREAK.

Read the help file for `list` and do the following:

3. Use list to show the values of the first twenty-five cases for the following variables: *socrel sexfreq age* and *sex.* See `help in` to see how to limit your results to the first twenty-five cases.
4. Repeat the listing above using the *`nolabel`* option and describe the difference between the two command results.

summarize

The `summarize` command calculates and displays a variety of univariate summary statistics. If no *`varlist`* is specified, summary statistics are calculated for all the variables in the dataset. There are few options and the output is very compact. Unless you have thousands of variables you can typically just enter summarize without the *`varlist`* option.

In the command window enter summarize without a *`varlist.`*

5. Based on this output, why can’t the variable year be stored as a byte? Why is it stored as an integer?
6. Revisit the variables *age*, *sex*, *race*, and *sexfreq* and revise, if appropriate what they measure.

tabstat

The `tabstat` command like `summarize` calculates and displays a variety of univariate summary statistics. The option *`varlist`* is required. A useful option is *`by(varname)`*. We will revisit this command but to see its value enter the following in the command window:

```
tabstat age, by(marital).
```

tabulate

The `tabulate` command produces one- and two-way tables of frequency counts. A *`varname`* is required. Typically when examining data you will use one-way tabulates but two-way ones can also be useful. A related command is `tab1` that allows you to `tabulate` several variables using one-way.

7. Use `tabulate` to describe the distribution of respondents sex. Repeat this command with the *`nolabel`* option. What can you see about the measurement of sex?

8. What is the difference between the following two commands:

```
tab1 sex race
tab sex race
```

9. Is the variable *socbar* missing for any years? Use the twoway version of tabulate to determine the years when this variable was not collected. Hint: In the help file for `tabulate` look at the *`missing`* option.

10. Use the `tabstat` command with the *`by(varname)`* option to determine what years *socbar* is missing.

11. Revisit the variables *age*, *sex*, *race*, and *sexfreq* and revise, if appropriate what they measure.

codebook

The `codebook` command is often used for documentation purposes and run once, saved in a file, and referred to as necessary. This command produces a lot of output for datasets with a large number of variables. To minimize the output you can specify a `varlist` or use the `compact` option which produces output similar to `summarize`.

12. Use `codebook` to describe the dataset in memory and log this output to disk.

Changing Data

Typically the data you begin working with will not be the data you end working with. As before, there are many ways to accomplish specific tasks in Stata. Here we will consider how to keep a subset of variables, a subset of cases, change values of variables, recode existing variables, create new variables, and label parts of our Stata files.

`generate` (`help generate`)

- Creates new variables

`replace` (`help replace`)

- `replace` change the values of a variable

`recode` (`help recode`)

- change the values of a variable

`label` (`help label`)

- Labeling variables, values, and data files

`notes (help notes)`

- notes can be attached to the dataset or
- to individual variables

`rename` (`help rename`)

- Change the values of a variable

`compress`

- reduces the size of a dataset by storing the variables as efficiently as possible
- changes the storage type if possible without any loss of information

`drop/keep` (`help drop`)

- Drop (or keep) one or more variables
- Used with if you can drop observations conditional on one or more variables or conditions

`edit` (`help edit`)

- Edit the data file directly (very dangerous; cannot be documented or replicated)

`browse` (`help browse`)

- `browse` is similar to edit, except that modifications to the data are not permitted

`generate`

The `generate` command is used to create new variables. There are many reasons why you might want to make new variables. You may want to recode a variable and first make a copy so you can retain the original. You may need to transform a variable. You may want to create an entirely new variable based on existing ones.

1. Create a copy of the age variable named *age1*.
2. Transform the age variable, creating *age2* equal to age^2.
3. Create a new variable called *educ1* equal to *educ* divided by 10.
4. Create a new variable called *sociability* that is the average of all of the sociability measures (*socrel, socommun, socfrend,* and *socbar*).

`replace`

The `replace` command changes the contents of an existing variable. It is often used in conjunction with `generate`. For example, you used `generate` to create the variable *age1*.

In this problem you will need to limit the replace command by a range of values. There are several ways to do this. You can use either the *`if`* option with replace or the `inrange` function.

5. Use `replace` with the *age1* variable so that cases with values between 18 and 31 are equal to 1; values between 32 and 43 are equal to 2; values between 44 and 59 are equal to 3, and values between 60 and 89 are equal to 4. These categories correspond to quartiles (determined using `tabstat`).
6. Check your work using `tabulate`.

`recode`

The `recode` command changes the contents of an existing variable and creates a new variable. You can easily add value labels with the `recode` command. Because of this I typically prefer `recode` to `replace` but both will change your data in predictable and reproducible ways.

7. Use `recode` to create a new variable, *age3* that uses the recoding scheme from the problem above (18/31= 1; 32/43= 2; 44/59= 3, and 60/89= 4). Be certain to add value labels for the age categories.
8. Check your work using `tabulate`. Also, `tabulate` *age1* and *age3*. What does this data check show you?

Data Annotation

There are many ways to make using a dataset easier—labeling things is one of those ways. In Stata you can give labels to 1) the dataset, 2) variables, and 3) values of variables. You can also attach notes to the dataset (e.g. the data source, current program revision, etc.) or to variables (e.g. "recode in response to 01/25/2010 meeting").

Review the online help file for `label` and `notes` and do the following:

9. Use `describe` to look at the current state of the dataset. Note that most of the new variables do not have a descriptive variable label. Use `tabulate` on the created variable *age1*. Note that there are no useful value labels. Compare this to *age3* that was created with value labels.
10. Add "Source: General Social Survey, 1972-2010" as a note to the dataset.

11. Add descriptive variable labels to *age1, age2, educ1,* and *sociability.*
12. Add the following value labels to the variable *age1*: 1="18-31", 2="32-43", 3="44-59", and 4="60-89". Define a *value label* called *age* and assign it to the variable *age1*.
13. Add a note to the dataset describing this assignment.
14. Add a note to the variable *age2* about the transformation.
15. Use the commands `describe` and `notes` to see your additions.

Cleaning Up the Dataset

After working on a dataset you may need to clean up your work. For example, if you didn't explicitly declare the storage type when using the `generate` command you may be storing data inefficiently. Are your variables named correctly? Are your variables in a logical order in the dataset? Have you kept only the cases and variables that you want?

Read the online help files on `compress`, `rename`, `move`, and `drop` and `keep` and do the following:

16. Use `compress` and describe that changes that were made.
17. Use `rename` to change the names of *age2* to *agesqr* and *age3* to *age2*.
18. Use `move` to put the age related variables in the following order: *age*, *age1*, *age2*, and *agesqr*; *educ* followed by *educ1*; and *sociability* before *socrel*.
19. Earlier we determined that the variable *socbar* was missing for several years. Eliminate all the cases for each of these years. Check you work with `tabulate` or `tabstat`. How many cases were dropped?
20. Reduce the number of variables in the dataset by eliminating *age1, age2, agesqr, educ1, sociability,* and *cohort.*

Create a Sociability Scale

Now we can do something useful—create a sociability scale based on the individual measures of sociability in the dataset. The measures (*socrel, socommun, socfrend,* and *socbar*) indicate how frequently people socialize. The exact wording of the survey items is:

Would you use this card and tell me which answer comes closest to how often you do the following things?	
socrel	Spend a social evening with relatives?
socommun	Spend a social evening with someone who lives in your neighborhood.
socfrend	Spend a social evening with friends who live outside the neighborhood.
socbar	Go to a bar or tavern.

All of these variables have the same category values and labels. Here is the frequency distribution for *socrel*. Note that smaller values imply *greater* sociability (1=”ALMOST DAILY” to 7=”NEVER”).

```
     SPEND EVENING |
    WITH RELATIVES |      Freq.     Percent        Cum.
-------------------+-----------------------------------
      ALMOST DAILY |      2,368        8.54        8.54
SEV TIMES A WEEK   |      7,514       27.09       35.63
SEV TIMES A MNTH   |      5,091       18.36       53.99
      ONCE A MONTH |      4,557       16.43       70.42
SEV TIMES A YEAR   |      5,080       18.32       88.74
       ONCE A YEAR |      1,929        6.96       95.69
             NEVER |      1,195        4.31      100.00
-------------------+-----------------------------------
             Total |     27,734      100.00
```

Create a variable called sociability that is an additive scale ranging from 1 to 25 with larger values indicating greater sociability. Label your variable and store it in the most efficient way possible. None of this is complicated but it does require some thought. For example, you need to reverse code all of the sociability variables so 1=NEVER and 7=ALMOST DAILY. If you simply add the four variables together the scale has a theoretical scale of 4 to 28—you need to adjust your scale to it goes from 1 to 25.

We will examine this variable in greater detail in future assignments so you need to get this right!

Finally, recode the variables sex and race to be 0/1 dummy variables where 0=male, 1=female and 0=white and 1=black. Be certain the variable is labeled and the values are labeled.

The following code creates a sociability scale variable that theoretically ranges from 0 to 100 where large values imply greater sociability. Try to decode how this code works:

```
generate sociability=round(((28-(socrel+socommun+socfrend+socbar))/24)*100,.1)
format sociability %9.1f
tab sociability
```

As a precursor for the next topic try the following commands:

```
graph twoway (hist sociability, discrete)
```