Creating and Naming Variables

*Note:* The creating and naming of variables is also an important part of writing documentation.

The fundamental principle for creating and naming variables:

*Never change a variable unless you give it a new name.*

The `generate` *(AKA gen)* command creates a new variable.

- **Almost ALWAYS** `generate = 0` …I have seen many mistakes made when people generate a new variable = .
  
  EX:  `gen newvar = 0`

- If possible, use a source variable when creating new variables—prevents the compounding of mistakes.
  
  EX:  `clonevar newvar = oldvar`

The `clonevar` command creates a new variable as an exact copy of an existing variable with the same storage type, values, and display format as the existing variable. Variable labels, value labels, notes and characteristics will also be copied.

Creating Variables

There are four simple principles:

1. If a variable is new, give it a new name

   EX:  Collapse the divorced and separated categories on variable `rmarital` into one category. Create a new variable named, for example: `rmarital_c` OR `rmaritalC` OR `rmaritalV2`

2. Verify that new variables are constructed correctly
   a. You can do this by running crosstabs of the new variable with the source variable(s) used to create the new variable

3. Document new variables with notes and labels (see subsequent sections)

4. Keep the source variables used to create new variables

Naming Variables

1. Use mnemonics—As discussed previously, a mnemonic naming system works best...**it is the easiest for our brains to work with.**

2. Try to use shorter names
   a. Stata allows for 32 characters, but most Stata commands show only 12 characters of a variable name, so...**Use names that are at most 12 characters in length.**
   b. Use capital letters sparingly, will give more meaning when you do use them (see next page for suggestions).
Letter | Meaning | Example
--- | --- | ---
B/D | Binary/Dummy variable | highsch1B
N | Negatively coded scale | menhlthN
P | Positively coded scale | phsyhlthP
V | Version # for modified vars. | marstV2
X | A temporary variable | Xtemp

c. Some datasets (e.g., NSFG) have variable names in ALL CAPS. Recommend you convert them to all lowercase.
   
   EX: rename *, lower

d. For household-level variables, I’ll create a suffix with the first letter of the HH id variable.
   
   EX: HH id variable is serial, all newly generated household-level variables will have a “s” prefix s_numbiokds

e. I did some mean substitution for my PAA paper. Because I had two different analytic populations—one for each of my dependent variables—I had to create new variables specific to each set of analyses.
   
   i. For the analyses predicting young adult coresidence with parents I appended the suffix “pc”
   
   EX: goodhlth_pc OR goodhlthPC

Label Variables
Every variable should have a variable label.

- Beware of truncation in output
- You can add notes to variables

EX: notes prtmart: div and sep are coded together
notes prtmart: source variable is marst

To see a variable’s notes type:

notes prtmart prtmart:
1. div and sep are coded together
2. source variable is marst

Value Labels
Assign text labels to the numeric values of a variable. Categorical variables should have value labels unless the variables has an inherent metric.

Principles for constructing value labels:

- Keep labels short: Variable labels should be eight or fewer characters in length
- Include the category number
You can include them in the syntax you type

EX:

```
lable define age1929_2c 1 "1. 19-23", modify
lable define age1929_2c 2 "2. 24-29", modify
lable value age1929_2c age1929_2c
lable variable age1929_2c "YA Age Cats."
```

You can also use the `numlabel` command. By running `numlabel`, add values will be prefixed to value labels of the variables in your dataset when run.

You can also run with a `mask()` option which controls how the values are added.

- `mask(#)` option adds only numbers (e.g., 1married)
- `mask(#_)` adds numbers followed by an underscore (e.g., 1_married)
- `mask(#.)` adds the values followed by a period and a space (e.g., 1. married)
- `mask([#])` adds the value in a bracket (e.g., [1]married)

EX: Prefix numeric values to repair value label using the specified mask

```
numlabel repair, add mask([#])
```

- Avoid special characters

EX: `.: = % @ { }`

- Apply vertically, NOT horizontally! For an explanation, please read the following Technical Note directly taken from the Stata help files:

**Technical Note**

Although we tend to show examples defining value labels using one command, such as

```
. label define answ 1 yes 2 no
```

remember that value labels may include many associations and **typing them all on one line can be ungainly or impossible.** For instance, if perhaps we have an encoding of 1,000 places, we could imagine typing

```
. label define fips 10060 "Anniston, AL" 10110 "Auburn, AL" 10175 "Bessemer, AL" ... 560050 "Cheyenne, WY"
```

Even in an editor, we would be unlikely to type the line correctly. The easy way to enter long value labels is to **enter the codings one at a time:**

```
. label define fips 10060 "Anniston, AL"
. label define fips 10175 "Bessemer, AL", add
...
. label define fips 560050 "Cheyenne, WY", add
Internally labeling documents
Principles for internally labeling documents (Word, Excel, etc.)

- Every document should include the name of the document file, author's name, and the date it was created.
  - In Word add a header (ensures information shows up on every page)
  - There is no header in a do-file, so the information should just come at the top of each do-file (see previous section on Writing legible do-files)
- Also include page numbers
  - In Word add to footer
  - In do-files, Stata does this automatically
- See this document as an example in Word.

Ex: Proper Header Information for an Excel File
Final Suggestions for Writing Documentation

Do it TODAY!
Check it later
Know where the documentation is
Include full dates and names

Ex: Do-File—Annotated

| Line 4. | Check the source variable by running a tab |
| Line 5-19. | Document the source variable with comments |
| Line 21. | Generate the new variable, giving it a new name |
| Line 23. | Add a comment regarding changes made |
| Line 25. | Give the new variable a label |
| Line 26 & 27. | Apply notes to the new variable |
| Line 29. | Check my new variable against my source variable to ensure it was coded correctly |
| Line 30-45. | Document the new variable check with comments |

Note: I truncated the tabs of the variables in order to get the crucial elements included in this annotated example.