# Basic Data Analysis Course on Stata

## Introduction to Stata

Md. Nayeem Chowdhury

Institute of Statistical Research and Training (ISRT)

December 14, 2024

# Labeling a data set

- label data

A data label is a description of the entire dataset. Dataset labels are displayed when you use describe command. It helps remind you what the dataset was about. For example,

label data /// “Data on Bangladeshi females at reproductive ages”

# Exploring/examining your dataset

It is a good idea to examine your data when you first read it into Stata.

- There may be unexpected data
- There may be missing data
- There may be miscoded data

# Creating new variables

- Sometimes we may need to create new variables those are function of some other variables.
- We can easily do that in Stata using function gen or egen.
- The former one is the basic command and the later one is an extension.
- The gen function is used to create a new variable which is a function of one or more other variables.
- The egen function has special functions such as mean, median, total, etc. that can be used to create a new variables.

# Essential operators

- We need to know some arithmetic, logical and comparison operators in order to subset the data or combine variables in Stata.
- The operators of Stata are listed below: Arithmetic Logical Comparison +————— ————- ————-
- addition ~ 0r ! not > greater than
- subtraction | or < less than
- multiplication & and >= > or equal / division <= < or equal ^ power == equal ~= not equal

- gen For example, we want to calculate the age difference between husband and wife for the BDHS data using variables v730 and v012. This can easily be carried out as,

gen aggap = v730 - v012 - egen Now, if we want a variable whose values are obtained by the mean age of women who belong to the same education group.

egen nage=mean(v012), by(v106) - gen ecb=0 replace ecb=1 if v212<18

# Recoding variable values

- Values of a variable can be recoded using function recode
- recode changes the values of numeric variable according to the rules specified
- Values that do not meet any of the conditions of the rules are left unchanged, unless an otherwise rule is specified.
- Basic form of the syntax is recode varlist (rule) (rule) . . . , generate(newvar)
- The most common forms for rule are

+———————————————————-+ | Rule | Example | Meaning | |————-+————-+————————| | # = # | 3 = 1 | 3 recoded to 1 | | # # = # | 2 . = 9 | 2 and . recoded to 9 | | #/# = # | 1/5 = 4 | 1 through 5 recoded to 4 | | nonmissing = # | nonmissing = 8 | all other nonmissing to 8 | | missing = # | missing = 9 | all other missings to 9 | +———————————————————-+

- This command changes the values of numeric variables according to the rules specified. For example,

recode v106 (2 3 = 2), generate(nv106) label define newlab 0 “no education” 1 “primary” /// 2 “secondary or higher” label values nv106 newlab - You can also categorize a numeric variable by using recode command and create a new variable using generate with a new value label defined within label. For example,

summarize v012 /*respondent's current age*/ recode v012 (15/17 = 0 “<18”) (18 19 = 1 “18-19”) /// (20/29 = 2 “20-29”) (30/39 = 3 “30-39”) /// (else=.), generate(agegroups) label(agegrp)

# Sub-setting data

keep

You can keep a subset of observations or variables for analysis and delete the others using this command. For example,

keep caseid v002 v003 v012 v024 keep if v012>=18 To keep the first five observations: keep in 1/5

drop You can subset the data by dropping some observations or variables using this command. For example, drop v730 drop if v012<18

# Sorting data

sort This command arranges the dataset according to ascending order of a variable. You can also specify more than one variable to sort.

sort caseid sort v190 v024 */division sorted within wealth index/* browse caseid v190 v024

# Sorting data

gsort This command stands for generalized sorting. This can arrange data both in ascending and descending order. You can just put a plus or minus sign before the variable to indicate ascending and descending order respectively. For example,

gsort +v190 */or simply: gsort v190/* gsort -v190 gsort v190 -v024 browse caseid v190 v024

# Reshaping datasets

- Sometimes it is needed to reshape data such as from wide to long and vice versa.
- How do you know if your data is in wide or long format?
- If there is only one unique case ID per row, then your data is in wide format.
- If there are several rows with the same case ID, then your data is in long format.
- Stata can do that simply using reshape command
- Now to convert the dataset from wide to long format we write: use faminc.dta, clear reshape long faminc, i(famid) j(year)
- To convert the dataset from long to wide format we write: use kids.dta, clear reshape wide age, i(famid) j(birth)

# Appending datasets

- You can combine different datasets into a single large dataset using the append and merge commands.
- First we create two separate data files by sub-setting based on age of respondents to obtain women aged less and greater than 18 years.

use bd_individual.dta, clear keep if v012>=18 save adults.dta count

use bd_individual.dta, clear keep if v012<18 save nonadults.dta count

# Appending datasets

- We can now use append to join these two datasets. The command append is used to add extra observations (rows) from another dataset to the existing dataset. append using adults.dta count
- Note that, you have to have the same number of variables in both data files before you append them.

# Merging datasets

- Now we create two datasets by sub-setting based on demographic and wealth information of the respondents. use bd_individual.dta, clear keep caseid v001 v002 v003 v010 v024 v025 v106 sort caseid save demographic.dta

use bd_individual.dta, clear keep caseid v119 v120 v121 v122 v123 v124 v125 sort caseid save wealth.dta - We have to sort the two datasets based on a common variable beforehand in order to merge them.

# Merging datasets

- We can now employ merge to join these two datasets. merge 1:1 caseid using demographic.dta
- Note that, the last column produced indicates the case was present in both master and secondary data.
- The open dataset is called the master data.
- For cases that come from only master data will show 1 in the column produced. For cases from only secondary data it will show 2.

# Merging datasets

- Now, we perform 1:m merge as follows: use data1.dta, clear merge 1:m household_id using data2.dta
- We can also perform m:1 merge using the following codes: use data2.dta, clear merge m:1 household_id using data1.dta