

**Handling Large-Scale Unit Level Data Using STATA**  
**Professor Pratap C. Mohanty**  
**Department of Humanities and Social Sciences,**  
**Indian Institute of Technology, Roorkee**  
**Lecture 18**  
**Tabulation and Creation of New Variables in Stata – II**

Welcome friends once again to this module of NPTEL on Handling Large-Scale Data Using Stata. We have been so far trying our best to give you the hands on experience with using the core dataset which we usually require for our everyday research. We require maybe for our PhD work, maybe for our paper and over last three, four lectures we have been trying to clarify the very basic commands of Stata which is usually very important for all the researchers who uses Stata.

So this lecture is a continuation of the previous one, where we have already discussed the prerequisite of unit level data in Stata. Where we discussed tabulation and creation of new variable with the help of Stata. In the last class particularly, I have shown you how to derive result based on the core dataset, results like I have explained you how to get tab and table results or tabulate versus table. The first and the very clear difference between these are to identify frequency as compared to percentages.

In table command I have already shown you that at best we can derive the frequency, only the frequency of the variables, whereas in case of tabulate or tabulation, we can get many more information as well, like percentages, even cumulative percentages, which are very important for researchers to start filtering the variables for further research.

(Refer Slide Time: 02:53)

## Three-way Tables for Categorical Variables

- You cannot give tab a list of three variables to act on.
- However, you can use **by** to create separate tables for each value of a categorical variable:  
`bysort var1: tab var2 var3`
- You can add as many variables as you need through **by** part of the command.
- However, the amount of output you'll get can become cumbersome quickly.

The screenshot shows the Stata command window with a keyboard overlay. The command entered is `use "G:\practice_dataset_MIS73_block2.dta"` followed by `tab b2_q204 Sector`. The output is a three-way table showing the distribution of ownership types across sectors.

Type of ownership code	Sector		Total
	1	2	
1	29,769	30,581	60,350
2	4,249	4,556	8,804
3	5	4	9
4	560	786	1,346
5	441	497	938
6	654	82	736
7	209	197	406
9	80	60	140
Total	35,744	36,742	72,486

Stata SE 11.1 - C:\practic\dataset\_N0371\prk02.dta

ownership code	1	2	Total
1	29,763	30,381	60,350
2	4,249	4,556	8,804
	49.25	51.75	100.00
3	5	4	9
	55.56	44.44	100.00
4	560	786	1,346
	41.60	58.40	100.00
5	441	497	938
	47.01	52.99	100.00
6	454	91	535
	84.86	15.14	100.00
7	209	197	406
	51.48	48.52	100.00
9	80	60	140
	57.14	42.86	100.00
Total	35,766	36,762	72,528
	49.31	50.69	100.00

Command

Stata SE 11.1 - C:\practic\dataset\_N0371\prk02.dta

ownership code	1	2	Total
1	29,763	30,381	60,350
2	4,249	4,556	8,804
	49.25	51.75	100.00
3	5	4	9
	55.56	44.44	100.00
4	560	786	1,346
	41.60	58.40	100.00
5	441	497	938
	47.01	52.99	100.00
6	454	91	535
	84.86	15.14	100.00
7	209	197	406
	51.48	48.52	100.00
9	80	60	140
	57.14	42.86	100.00
Total	35,766	36,762	72,528
	49.31	50.69	100.00

Command

The screenshot shows the Stata interface with a data table. The table has columns for 'Sector' and 'row' percentages. A keyboard overlay is present in the center. The 'Variables' window on the right shows the structure of the 'Sector' variable.

Sector	row %
1	49.33
2	49.25
3	55.56
4	41.60
5	47.01
6	84.86
7	51.49
8	57.14
9	57.14
Total	49.31

The screenshot shows the Stata interface with a two-way table for 'Sector = 2' and 'Nature of operation'. The table shows counts and percentages for 'perennial', 'seasonal', and 'casual' categories. The 'Variables' window on the right shows the structure of the 'assistance\_rec' variable.

Nature of operation	Y	N	Total
perennial	447	36,017	36,464
seasonal	3	222	225
casual	1	71	72
Total	451	36,310	36,761

Two-way table we already explained. To explain those things, I will show you once again how we operated the two-way one, then only we can discuss the three-way table. The two-way table, what does it mean? It is also called cross tabulation, two variables how they are related, maybe they are correlated, maybe one is dependent variable, one is independent variable and how you wanted to establish the relationship between two variables the very basic two-way table can able to clarify it.

So let me just open one dataset for you which we wanted to use, is the following. I have explained you for the NSS 73rd in the last class. Let me open for you once again. So the two

table we wanted to show and its relationship through the tabulation command is simply typing `tab` or only `ta` that I have shown you in the last class. I think in the last class I have shown you already how to go for it. And like this let me proceed. Here one variable I wanted to use is type of ownership `b2_q204` with that by sector. `Tab`, so I will start with type of ownership here. Then I will also use another variable since I am interested now to discuss two-way table. So let it be sector here.

For a simple result without any percentages, we can simply enter. We will get the frequency with two variables and their cross relationship between these two variables. How each of the variable with their different codes are overlapping, like 1 here, then 1 there, and 1 stands for, I think already I clarified, 1 is rural and 2 for urban. Similarly, in the type of ownership, we have already discussed proprietary male, proprietary female, then proprietary maybe by joint venture, proprietary by SHG Group. These are very different code. You can find it from the original data from their layout also. We will also explain those things in a different format today. So this is one way we have discussed in the last class.

Just frequency and its number is cumbersome, in the sense that you may not get the right picture for analysis. You can at best say that 29,769 number are from rural areas and those were the male proprietor out of 72,528 number of unorganized enterprises as per the 73rd round of National Sample Survey, Government of India data. But this number probably you cannot compare with another number because these are not in percentages, these are in absolute number. You always require a percentage figure for better comparison.

So percentage you can derive, as I told you, by giving another addition to the command. Suppose you are interested for row wise percentages or column wise percentages; you need to add with another command to it that is `row`. So, `row`, so if you add it, it gives you the information like row wise percentages. Similarly, I have shown you all those things. So I am not going to spend much time today. You can have a revisit.

But one thing I just want to mention here, if you are not interested to come out with your result with the absolute frequencies, you only want to keep your result with their percentages, so in that case `nofr` is the right command. So you simply click there, no frequencies result, `nofr`. So row wise `nofr` or column wise `nofr` both. If you wanted to go by row wise, it will give you row wise

or if you add both row and column with `nofr`. But you have to identify whether it is row wise or whether it is column wise. But if you want both, then you have to add row and column both. So if I add, it gives result like this, row wise as well as column wise.

Another aspect of looking at the data and its interpretation is through cell percentages. I think I have told you in the last class that these are the figures derived either by row total or by column total, but what about the absolute total, individual cell, single cell. I told you in the last class, single cell out of the total. So, in that case, you simply add `cell` command and that will give you the right result.

Let us come back to the three-way table. Two-way we discussed even some other additions to the two-way table and their interpretation we mentioned at large in the last class. Let's start with the three-way table. In the case of three-way table, why it is essential for a researcher. The three-way table gives some kind of comparison between two variables with respect to a third variable. The third variable when you are doing, it is always suggested to go by a sorting of the third variable. It sorts the third variable. If want to focus a particular variable, you start with that particular variable. It will sort with that particular variable in addition another two variables will be added.

In this case, three-way tables for categorical variable. One important aspect to be mention, when it is a scale variable or a continuous variable, this table can derive result, but it is very difficult to read, because scale, there are so many rows and columns will be derived in case of two-way or three-way comparison. So, Stata might tell you that there is too many information, so better to stick to categorical variable.

So, categorical variables limit your coverage of the data to particular group. So, the group comparison we have already shown it here, as given the variable type of ownership we have nine groups and regarding sector we have two groups. So, comparison is clearly visible and is understood better. But in case of continuous variable, it is too difficult to do it. And though we get the result, but interpretation is not going to be so apt.

So you cannot give `tab` a list of three variables to act on. Looking at three variables just `tab` three variables is not going to give you any result. So in that case, the right command is through `by` command. We would have to separate it through a command called `by` for each of the categorical variable. So we have given here with this right command, `bysort` of a particular variable, as I just

mentioned that you need to sort that variable and the right command is `bysort` of the particular variable then you add colon then tab, likewise we did for the cross tabulation.

`Bysort` variable, in this case let me clear this data and open another one for you. Now I am going to open the recoded data we have simplified already. I think this is the one. So recoded data where all the variables with their code, different names have been recoded and properly mentioned for our easy use. There are, like here, that time I could not able to read correctly. Later on I just read by this second labeling information, but the name was confusing.

Here I can read easily if they are registered or not, growth status or not, whether the particular enterprise has received assistance or not, whether the enterprise is using Internet or computer etc. nature of operation information also there. So it will be very easy to use. So let me use the command `bysort`. I need to look at the enterprise, I will sort it by sector but I will compare the status of the enterprise over last three years, where is that variable, nature of operation variable which just used.

So, nature of operation of the enterprise with respect to their assistance received. So I will compare whether nature of operation is having any relationship with assistance received, but I will sort it by sector. So I will start with `bysort` then I will start with that variable first by sector, isn't it? So I have added sector here. Then I will have to add a colon. This is a semi-colon, but I will have to add colon. Then the normal cross tabulation is going to be useful. Just `ta` is enough for us. What I wanted to look at how the nature of operation of the enterprises here is looked at through their assistance received. I think assistance received is given here. I will add, let it be by row wise percentage. So row wise percentage is here. After doing so, I derived the result like this.

So, it has divided the result into two parts. You just go little up. You will find. Here, it has divided into sector one. At the bottom it is sector two. At the top, it is sector one. So sector one in rural areas what is the relationship between their nature of operation, perennial, seasonal, casual. And in the urban areas how these are looked at through the context of whether the enterprise received assistance from the government during last three years. Sometimes we are interested to interpret through assistance received. If it is assistance received through government, we define them as formal assistance, formal credit.

So whether those enterprises who operate perennially or seasonally or casually, the total number of enterprises, whether they have received assistance or not. Look at the interpretation here. Out of 36,761 enterprises in urban areas, out of that we have divided into perennial, seasonal, we have derived row wise, first division is through assistance received or not received. 451 reported that they have received assistance, whereas 36,310 reported that they have not received any assistance.

Those operate perennially out of 36,461 in the urban area, only 1.23% reported having assistance from government sources, financial assistance from government sources. So is not it interesting for the researcher to note. Similarly, seasonally it is 1.33, casually that is 1.39. When you have already defined your variable very correctly, your interpretation is going to be very good. I am just showing you randomly how this is impacting our result and how you can derive the result.

So let us move on to our next set of explanation those will be going to be very interesting. So you can add as many variables as you need through by part of that particular command. However, the amount of output you will get can become cumbersome quickly if there are so many variables. So it is suggested that you stick to three variables. More variables generally there will be so many layers, sub-layer, then sub-layers, it will be too difficult to interpret.



(Refer Slide Time: 18:32)

### ❑ 3-way, 4-way, 5-way or More Tables in Stata

Table is the command with which you can produce such tables in stata.

```
table row_variable column_variable super_column_variable,  
by(super_row_var_list) contents(freq)
```

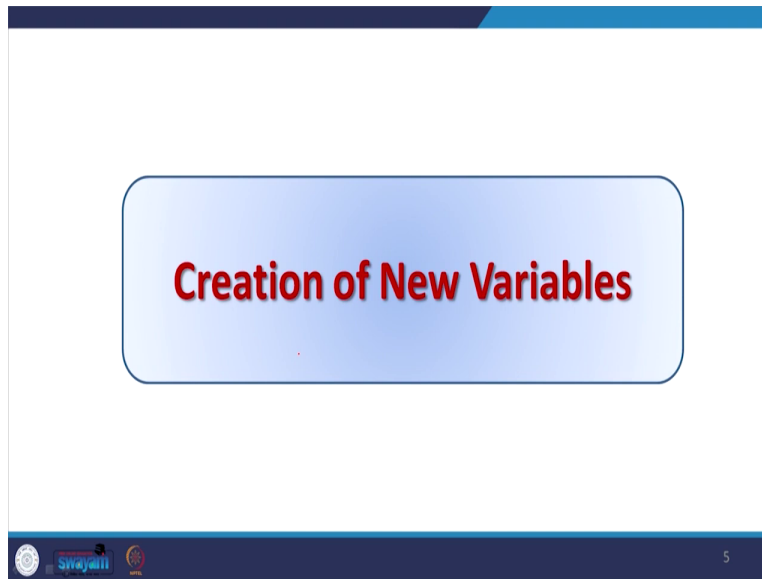
Note: the third variable we use with table command is called super column variable. The list of variables attached with by option are called super row varlist.

So, three-way, four-way or five-way or more tables in Stata is always possible. The table command which I have already told you gives you the right result very quickly. Like table is the command with which you can produce such tables with the help of Stata. The simple command like table which simply give the variable's name, it gives all the results.

How we read those variables? The first variable is row variable; the second variable is column variable. Then the third variable whatever you have entered is called super column variable and the fourth one is called by variable, super row variable list. The variable you simply command, but the interpretation is generally by the way we interpret, it is by that command only you have to add by, there only. Without by it is not going to read that.

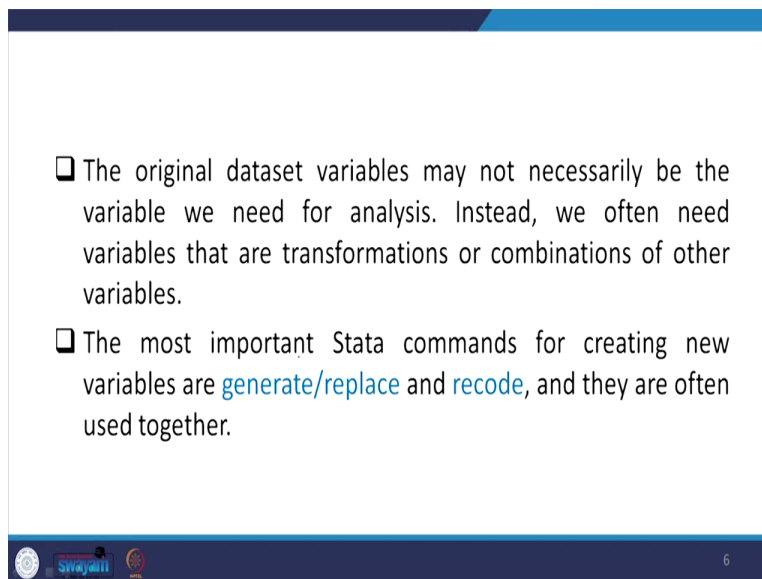
Then content with its frequency if you add, it gives the right result. One thing you need to note is that the third variable we use with table command is called super column variable. This, the list of variable attached with by options are called super row variable list, which is already underlined here or mentioned here. But it is always important to note that this gives frequency, not the percentages. So generally researchers use less table as the command.

(Refer Slide Time: 20:36)



Let us come to the interesting part of our lecture called creation of new variables. How to get new variables, create new variables with the help of Stata commands.

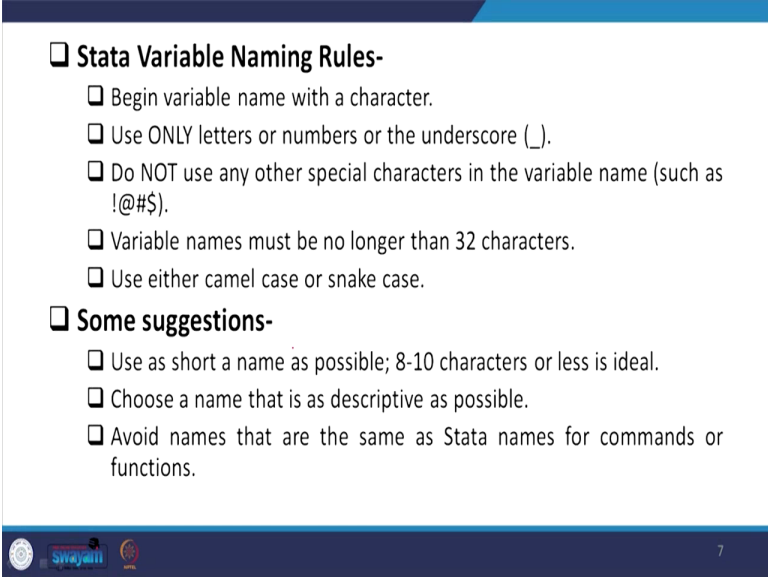
(Refer Slide Time: 20:49)



The original dataset variables may not necessarily give you what you need in the analysis. In this case, we often use or need variables that are transformations or combinations of other variables. So the original variable only gives the overview. For our analysis we need to transfer or combine the variable for our better interpretation. The most important Stata commands for those creation

of new variables are generate, replace, recode. We have already discussed replace and recode earlier, and we are discussing in the context of generating a variable. They are often used together, all those commands usually taken together for deriving a new variable.

(Refer Slide Time: 21:41)



**❑ Stata Variable Naming Rules-**

- ❑ Begin variable name with a character.
- ❑ Use ONLY letters or numbers or the underscore (\_).
- ❑ Do NOT use any other special characters in the variable name (such as !@#%).
- ❑ Variable names must be no longer than 32 characters.
- ❑ Use either camel case or snake case.

**❑ Some suggestions-**

- ❑ Use as short a name as possible; 8-10 characters or less is ideal.
- ❑ Choose a name that is as descriptive as possible.
- ❑ Avoid names that are the same as Stata names for commands or functions.

7

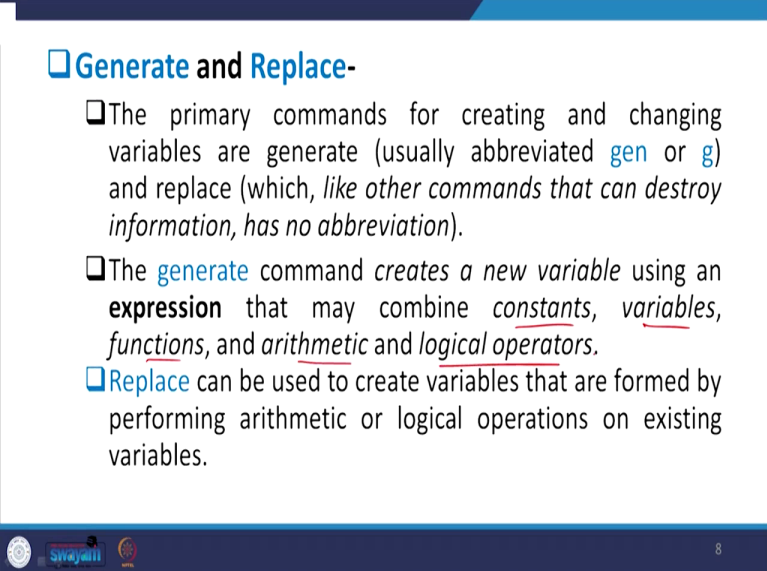
Some important rules behind creating new variables in Stata, so you begin the variable name with a character always and use only letters or numbers or the underscore. So, space in between two names or character is not going to be read by Stata. You have to either go by underscore or the snake based variable design you can also make it. You can start with a new name or new character with a capital letter. So that it can also read.

So another important rule is, do not use any other special characters in between or in the variable name such as exclamation mark or at the rate, hash or dollar. Those should be avoided, because Stata is not going to read these, not going to accept as the variable name. The variable names must be no longer 32 characters. It should not exceed 32 characters in the variable name.

So use either, as I just mentioned, snake based or camel based. Camel based as I told you, do not want to keep the space, you can start with a capital letter in the next separated name, but it has to be continuous with a higher capital letter. But in case of snake you need to give a space in between.

There are some suggestions. Use as short as possible for the name, usually 8 to 10 characters or less is the best way of naming the variable. Choose a name that is as descriptive as possible. Avoid names that are the same as Stata names for commands or functions.

(Refer Slide Time: 24:07)



**Generate and Replace-**

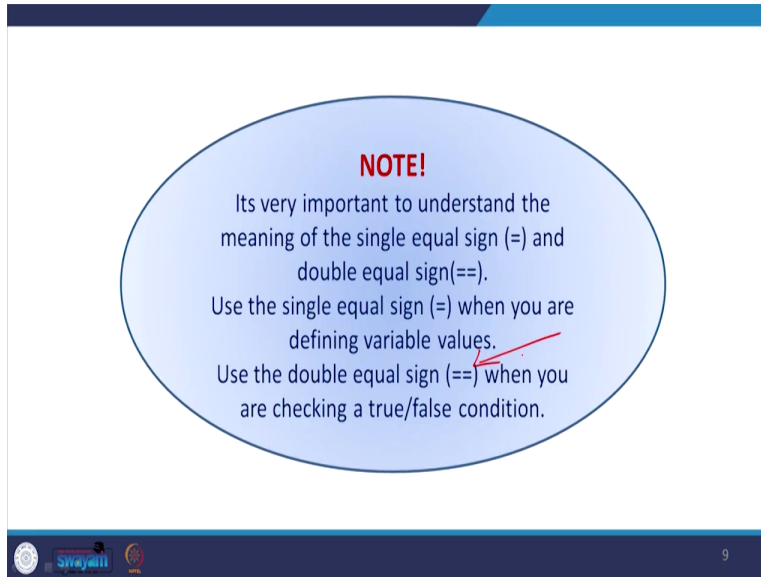
- ❑ The primary commands for creating and changing variables are generate (usually abbreviated **gen** or **g**) and replace (which, *like other commands that can destroy information, has no abbreviation*).
- ❑ The **generate** command *creates a new variable* using an **expression** that may combine constants, variables, functions, and arithmetic and logical operators.
- ❑ **Replace** can be used to create variables that are formed by performing arithmetic or logical operations on existing variables.

8

So in order to understand the variable, we are supposed to go by generate and replace and there are specific commands for it and the primary commands for creating and changing variables are generate. Usually, we abbreviate it with gen or g with a comma replace. Replace is important because the earlier any kind of entries are there with the same name or even any database is there, it will not be confusing further. So which like other commands that can destroy the information and has no abbreviation like this.

So generate command creates a new variable using an expression that may combine constants, variables. We are going to discuss all those things, functions, arithmetical, logical operations. This is there in our slides we are going to show it. Replace can be used to create variables that are formed by performing arithmetic or logical operations on existing variables. We are going to show it.

(Refer Slide Time: 25:00)



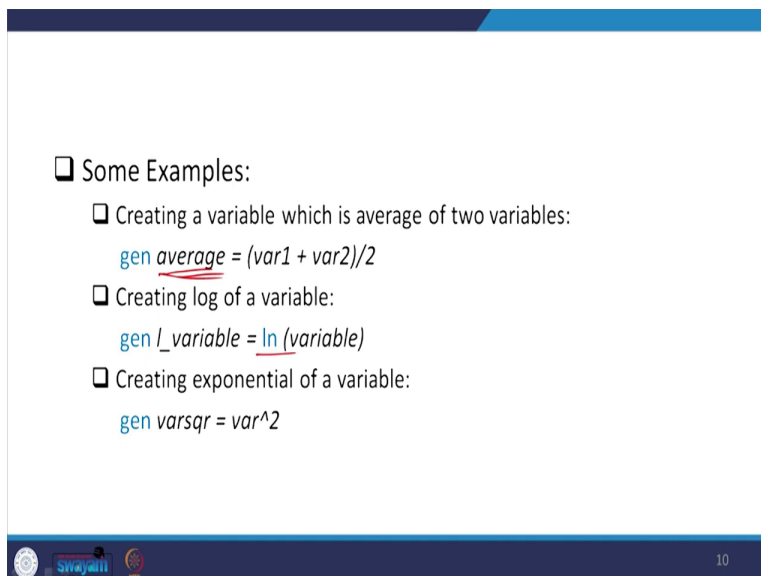
**NOTE!**

It's very important to understand the meaning of the single equal sign (=) and double equal sign(==).  
Use the single equal sign (=) when you are defining variable values.  
Use the double equal sign (==) when you are checking a true/false condition.

9

It is very important to note that you should understand some of the meaning of equal sign and double equal sign, like the meaning of the single sign and double sign we are going to tell you. Use the single equal sign when you have to define variable value, whereas the double equal to sign, like here it is mentioned, the double equal to sign in the command if you are giving it while in the generate command when you are going to check a true or false function.

(Refer Slide Time: 25:45)



❑ Some Examples:

- ❑ Creating a variable which is average of two variables:  
`gen average = (var1 + var2)/2`
- ❑ Creating log of a variable:  
`gen l_variable = ln (variable)`
- ❑ Creating exponential of a variable:  
`gen varsqr = var^2`

10

Stata/11.1 - O:\example\_data\_set.dta

```

1 use "O:\practice dataset N..."
2 ta.b2_q104 Sector
3 ta.b2_q104 Sector, row
4 ta.b2_q104 Sector, row nch
5 ta.b2_q104 Sector, row col
6 clear
7 use "O:\example data set.d..."
8 bysort Sector ta.nature_rpr
9 ta.growth_status

```

Review

Filter commands here

Command

ta.growth\_status

	personnel	seasonal	casual	Total
447	36,017	36,464		
1.23	98.77	100.00		
3	222	224		
1.33	98.67	100.00		
1	71	72		
1.39	98.62	100.00		
<b>451</b>	<b>36,310</b>	<b>36,761</b>		
1.23	98.77	100.00		

ta.growth\_status

Status of the enterprise over the last 3 years	Freq.	Percent	Cum.
expanding	24,324	35.18	35.18
stagnant	28,883	41.77	76.95
contracting	6,558	9.48	86.44
	9,378	13.56	100.00
<b>Total</b>	<b>69,143</b>	<b>100.00</b>	

Variables

Name	Label
enterprise_ty	Enterprise type (du...
netval_year	Year of net val...
nature_rpr	Nature of operation
accounts_ma...	Whether accounts...
computer	Did the enterprise...
internet	Did the enterprise...
prob_facid	Did the enterprise...
assistance_rec...	Did the enterprise...
growth_status	Status of the enterp...
registered	Whether register...
contract	Does the enterprise...
sovet_facility	Does the enterpris...
NSS	number of first sta...

Properties

Variables

growth\_status

Name: growth\_status

Label: Status of the enterp...

Type: byte

Format: %10.0g

Value label: growth\_status

Notes

Data

Filename: example\_data\_set.dta

Label:

Notes:

Variables: 50

Observations: 72,518

Size: 17,294

Memory: 64K

Sorted by: Sector

Command

Stata/11.1 - O:\practice\_data\_set.dta

```

1 use "O:\practice dataset N..."
2 ta.b2_q104 Sector
3 ta.b2_q104 Sector, row
4 ta.b2_q104 Sector, row nch
5 ta.b2_q104 Sector, row col
6 clear
7 use "O:\example data set.d..."
8 bysort Sector ta.nature_rpr
9 ta.growth_status
10 clear
11 spuse lfexp
12 gen avg_gppcp = (popgrovth + gppcp) / 2

```

Review

Filter commands here

Command

ta.growth\_status

	personnel	seasonal	casual	Total
447	36,017	36,464		
1.23	98.77	100.00		
3	222	224		
1.33	98.67	100.00		
1	71	72		
1.39	98.62	100.00		
<b>451</b>	<b>36,310</b>	<b>36,761</b>		
1.23	98.77	100.00		

ta.growth\_status

Status of the enterprise over the last 3 years	Freq.	Percent	Cum.
expanding	24,324	35.18	35.18
stagnant	28,883	41.77	76.95
contracting	6,558	9.48	86.44
	9,378	13.56	100.00
<b>Total</b>	<b>69,143</b>	<b>100.00</b>	

Variables

Name	Label
region	Region
country	Country
popgrowth	Avg. annual % gro...
avg_gppcp	Life expectancy at...
gppcp	GDP per capita
lfexp	Life expectancy at...
avg_gppcp	

Properties

Variables

avg\_gppcp

Name: avg\_gppcp

Label: GDP per capita

Type: float

Format: %10.0g

Value label:

Notes

Data

Filename: lfexp.dta

Label: Life expectancy 1998

Notes:

Variables: 7

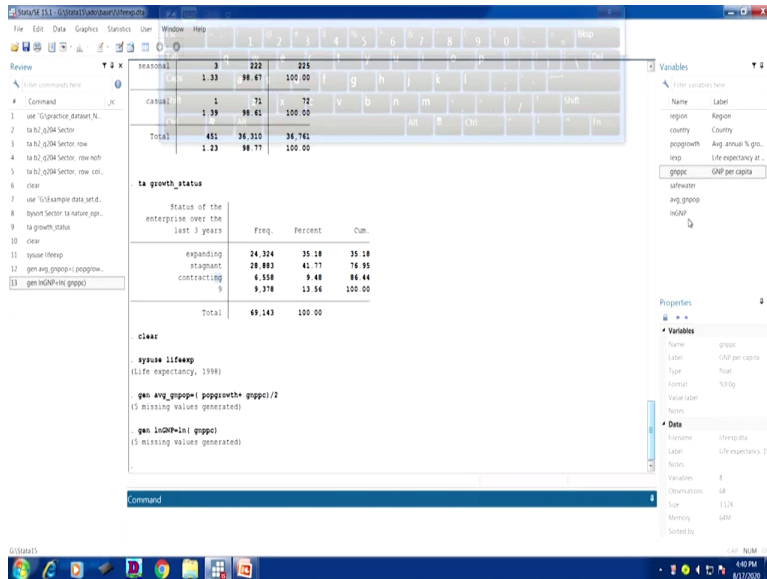
Observations: 68

Size: 28K

Memory: 64K

Sorted by:

Command



Like you can usually do that. I will also show you. Like creating a variable which is average of two variables. You wanted to get the average of two variables and you wanted to create another variable with average of two variables. So simply `gen` you can write and take a name, but better to combine with average underscore you can also take the name of that or simply name, this stands for a name, better to give a right name, because you are going to operate with average. So click the variable one or type the variable one plus variable two, close the bracket and divided by two. Slash two takes as divide. So it will show you.

Let me read out all those things and I will operate. Let us go by one example and in this particular data like you wanted to take the average of two variables, you have a continuous variable. If it is numeric one it will be good to average, but having categorical information averaging is not suggested unless the reason of averaging and your interpretation for it. So, better to stick to a continuous variable, for example, growth status. What is there in the growth status let me first check for you. So I simply wanted to check, if I just type like this growth status, it gives me information like it is also categorical: expanding, stagnant or contracting.

Look at, there is a nine entries. I have already discussed that, if any other entries are there, nine, triple nine, these as per the NSS data, this suggest that you please read those as missing. We have specific commands. Like number, what is this, number of first stage units. Let me just operate any two variables and its average. You can go by your own way. Otherwise, I will simply go by

the Stata directory. The directory it has better information. So let me clear this first, then I will start with the system use data.

Sysuse, then as I told you, lifeexp data we have. here population growth is there, then life expectancy is also there, you wanted to generate a variable. You simply write down gen you should take the name of that particular variable, let me take the name as avg, average underscore gnp and population growth. gnp per capita and population growth, just I wanted to take the average. How these continuous variables go by average number. So let it be average gnp, I am just using gnpop is equal to then bracket start, shift, then two variables, I have already said this plus this then bracket close then divided by two. So if you enter it, it defined you already, another variable with average genpop and it shows that there are five missing values generated. We will also discuss why missing values are generated in a short while.

Similarly, if you wanted to discuss about converting the variable into log of it, like usually the consumption expenditure or the income or the wealth type of variables are required to be converted because usually those data are not normal in nature. Those are usually skewed. Income of the population in the country is skewed. It skewed towards only 1% of or only 10% of the population occupies 90% of the total income in our country as per some of the reports. That means it is skewed.

But when you have used the skewed variable for your interpretation, it always gives biased result, because you have not scale down the variable, you have not made the variable normal. It will again bias your variable and its interpretation, other variable interpretation. So in that case you need to scale down the variable with any form of scaling. One important scaling down of that particular variable is through logarithmic transformation.

So generate, here we have used log of it, log of that particular within bracket of that variable, but I have used the gen, like in this case gnp per capita income is a continuous variable and it seems that it is not a normal variable, you need to scale it down. Then go by gen, you have create a variable, but how do you identify this is different than that of the original variable you have to attach with l somewhere, l stands for logarithm. You can say ln if it is natural logarithmic, ln you can underscore to it or you can simply write down capital GNP then just simply type equal to



then ln, since we are taking the natural logarithmic function here, ln within bracket this variable then you close that bracket.

This variable is going to be converted into a proper scaled variable and it can be interpreted and used for further analysis. This is created. Here the variable has already been created as log gnp data. Similarly, for if you wanted to square a variable or cube a variable you can use the power, to the power two. Then that will give you a variable for sure. So I am not going to explain all those things. You can experiment. If there are any possible doubts, please note it down and ask to us. We will clarify for sure.

(Refer Slide Time: 33:54)

□ The `replace` command modifies existing variables in exactly the same way as `generate` creates new variables:

```
gen l_variable = ln (variable)  
replace l_variable = ln (1) if l_variable =.  
replace yr=yr-100 if yr >= 100- meaning it replaces 0,1,etc  
instead of 100,101 etc.
```

So, similarly, the `replace` command modifies existing variables in exactly the same way as `generate` creates new variables. Basically, here we are going to replace the variable with another way. Like `generate` variable we did log of that variable and log of this. Here, similarly, `replace` you take a name of it then log one. Basically why `replace` is important, I have already clarified in a point in the previous page in our slide that if one variable is by any reason turned out to be dot, like we converted some variable, but those are like I have taken log of it, but that has been converted as dot. How you can replace that to another one, because if you have missing values, it might be reducing your number of cases for the answer, for the interpretation or for the result. That may weaken your result.

Or like some variable for logarithmic transformation you have already an entry with dot, a lot of dot is not going to give you any result. So in that case try to change that to one, simple one. So lot of one is equal to zero. So it will convert a variable and at least the logarithmic function will be operated. So in that case you simply replace that variable with one, log of one. It will change that particular dot to one and it will operate your function.

Similarly, replace is useful. Look at the difference between replace and generate. Replace year but it has to be if it is till 100, if year is greater than equal to 100 that means it replaces zero one etc. instead of 100, 101 or any other value. So like if you are very clear with what you want 100 to be replaced and it replaces only zero and one, not above to that of 100, 101 accordingly. You can just go through and you will find out for sure. It is not that difficult and you can easily able to do it.

(Refer Slide Time: 36:34)

**Operators and Expressions-**

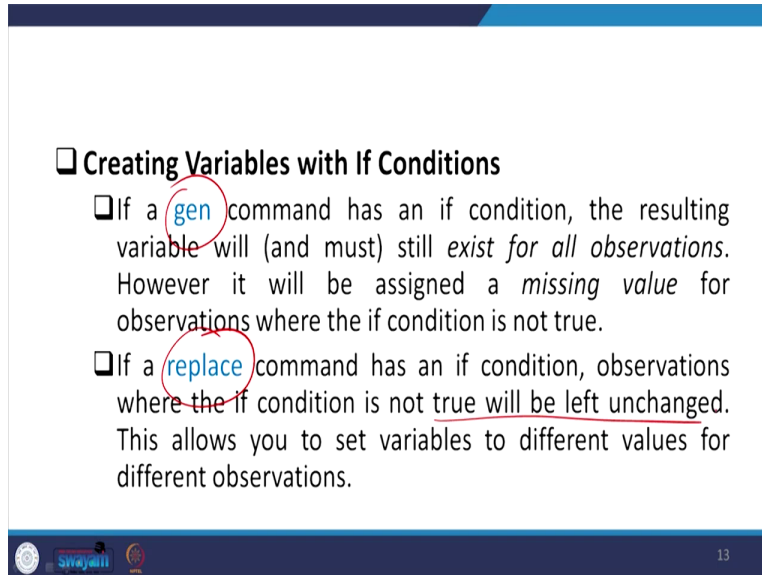
- Arithmetic**
  - + add/ string concatenation
  - Subtract
  - \* Multiply
  - / divide
  - ^ raise to power
- Logical**
  - ! not (also ~)
  - | or
  - & and
- Relational**
  - == equal
  - != not equal (also ~=)
  - < less than
  - <= less than or equal
  - > greater than
  - >= greater than or equal

12

Let me further clarify what are other aspects of operations and expressions we need to do it. There are broadly arithmetic operations, logical operations and relational operations. So arithmetic operation add string, concatenation, subtract, multiply, divide, like power operations, the exclamation mark, not equal to or, and in logical operations, similarly, relational double equal to or exclamation with equal to that does mean not equal, please go through this you will clarify and try to operate the way we have entered logarithmic transformation or some greater than

transformation. You try to operate and use these for your explanation. You will certainly have some doubts and we will be happy to clarify those later.

(Refer Slide Time: 37:46)



**□ Creating Variables with If Conditions**

- If a **gen** command has an if condition, the resulting variable will (and must) still *exist for all observations*. However it will be assigned a *missing value* for observations where the if condition is not true.
- If a **replace** command has an if condition, observations where the if condition is not true will be left unchanged. This allows you to set variables to different values for different observations.

Let us come to creating a variable with if condition. If a generate command attach with an if condition or if a gen command has an if condition, the resulting variable will still exist for all observations. However, it will be assigned a missing value for observations where the if condition does not follow. You have an if condition and we have an example here. We will clarify accordingly.

Similarly, if a replace command has an if condition, observations where the if condition is not true will be left unchanged. One difference between gen and replace is that, in gen you are converting that particular variable. You are creating a variable with a new one. But in that case if your if condition is not right, that will generate a missing value. But in case of replace one, that will be left unchanged. That is very very important to be noted. This allows you to set variables to different values for different observations. We have example here. Let me go by that and clarify you.

(Refer Slide Time: 38:54)

□ In our example dataset, we have a variable called type of ownership of an enterprise which has 8 categories. If we want to create a new variable with only male entrepreneurs:

```
gen maleEntrepreneur = 1 if ownership_type < 2 or == 1
```

This command generates variable with values 1 for ownership type less than 2 or equal to 1 or the other gets missing.

```
replace maleEntrepreneur = 0 if ownership_type > 1
```

this command changes those missing to zeros.



14

In our example dataset, we have a variable called type of ownership, which I have already shown you, which has eight categories. If we want to create a new variable with only male entrepreneurs, so in that case generate male entrepreneurs, any name we will give, equal to if ownership type, that is the variable name given in the data, less than two or double equal to one. So ownership less than two or double equal to one, what does it mean. You wanted to create a male entrepreneur as one. These commands generate a variable with value one for ownership type less than two or equal to one or the other gets missing. Basically it is less than or two since we have limited already with one or less than two, if there are any other entries in the ownership type, there are other entries already I have shown you, so those will be converted into missing dot. Then later on we have to give another command for filling those dots.

But the replace command, if are going by the replace command with zero, like replace male entrepreneur equal to zero if ownership type is greater than one. Like all ownership type which has the value greater than one, it will be replaced with zero. So, these commands changes those missing to zero. Even if missing is there that will be also converting to zero. But generate is not going to do that. You can just practice, you will certainly get it. I have many aspects to deal with. So I am emphasizing other details as well.

(Refer Slide Time: 40:57)

### □ recode-

- Variables are often not coded the way we want, often with too many categories or with values out of order. With `recode`, we can handle all recodes for a variable in a single step.
- The core of the recode command is a list of rules, each in parentheses, that tell it how a variable is to be recoded.
- They take the form ***(inputValue = outputValue)***.



So another, I will just clarify, let me come to it. I will clarify this recode and then after recode we will take another lecture for another one. So, for recoding, variables are often not coded the way we want, often with too many categories or with values out of order. So with recode, like if it is out of order, the way you want to recode it is not there, so we can handle all recodes for a variable in a single step, simply recoding. We are going to show it. The core of the recode command is a list of the rules, each in parenthesis that tell it how a variable is to be recoded. They take the form like input value to output value. We will generate with a output with that particular recode. I am going to tell you the exact example.

(Refer Slide Time: 42:06)

### Note!

The *inputValue* can be a single number, a list of numbers separated by spaces (1 2 3 = 1), or a range of numbers specified with *start/end* (1/3 = 1). Output value will always be single number.

recode location\_enterprise (1 = 1) (2/6 = 2), gen (location)

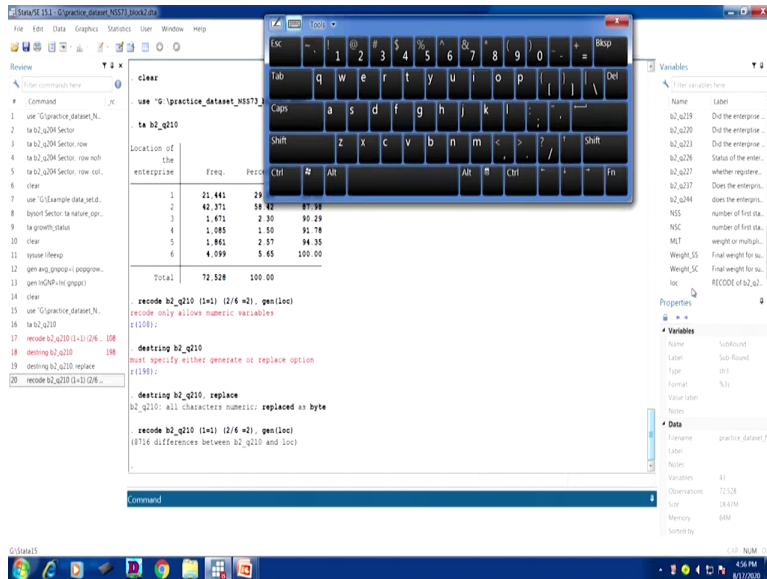
The above command will recode the location of enterprise which has 6 categories into two categories only.

The screenshot displays the SPSS Command Syntax Editor with the following commands and their output:

```
1 use 'G:\practice_dataset_N_1031_block2.dta'
2 . clear
3 use lifeexp
4 . recode lifeexp (1=life expectancy, 1998)
5 . gen avg_gppppi = (gpppcw + gpppc) / 2
6 . clear
7 use 'G:\Example_data_set.dta'
8 bysort Sector : ta nature_op_
9 ta growth_status
10 . clear
11 use lifeexp
12 gen avg_gppppi = (gpppcw + gpppc) / 2
13 gen loc = ta location_enterprise
14 . clear
15 use 'G:\practice_dataset_N_1031_block2.dta'
16 ta loc_210
```

The output shows a frequency table for the variable 'location of the enterprise':

location of the enterprise	Freq.	Percent	Cum.
1	21,441	29.56	29.56
2	42,371	58.42	87.98
3	1,671	2.30	90.29
4	2,985	4.15	94.44
5	1,861	2.57	97.01
6	4,099	5.65	100.00
Total	72,328	100.00	



The input value can be a single number. A single number, a list of numbers separated by spaces 1, 2, 3 is equal to 1, like there are codes given already 1, 2, 3, we want to recode it to 1 or a range of numbers specified with start to end. Like if we have a continuous series and codes are given like maybe 1 to 8 codes, in our ownership case, 8 codes are there, 1 to 8. You wanted to only code with converting those to 1 to 5 as 1, for example. So instead of writing 1, 2, 3 equal to 1 you simply write down 1/3 or in our case 1/5 equal to 1 it will recode. So output value will always be a single number that is 1, isn't it?

So in our example you can operate accordingly. So recode with that location enterprise that is, recode location underscore enterprise within bracket 1 is equal to 1. Basically you have kept 1 as 1, but others as 2. 2 to 6 as 2, you have given a code like 2/6 is equal to 2 then you can generate that variable, suppose you are recoding location, recoded you can also take a name redefined or location, another name you can define according to your own choice, we have given the name here as location.

I can recode for here like we have now, I have to go by that example the enterprise example. I can also recode that country, coding, I have to check that. It is probably problematic. Let us go by the variable. Let me clear first. Then I will open another one. I will open our original data. Then I will operate. So it is there.

If it is there, even from this also, I have already shown you, nature of operation is here, enterprise type, location, nature of operation, so here we wanted to show you location of that enterprise. So

let me check what type of codes are there in the location of that enterprise. I wanted to, there are six codes in the location of the enterprise given. I wanted to keep 1 as the code remain intact. For others I wanted to code it another one as 2.

So what I will do, I will go by the command recode then I have to discuss this location of the enterprise. I have to get this variable here. Then within bracket, what I will do, I will define as, there are 1 to 6 code, 1 as 1 that is basically within the household operation and outside the household operation. 1 stands for within the household and rest broadly categorize as outside household operations so far as enterprise operations is concerned.

1 equal to 1, isn't it, then bracket close, then within bracket another one, then either I will do one thing, I will do separate two, there are so many ways of doing. 2 is equal to 2, 3 is equal to 2, 4 is equal to 2, 5 is equal to 2, 6, there are so many ways of handling or in simple term 2, 3, 4, 5, 6 equal to 2 the way we have already shown you that we can do it or in simple way we will simply do 2/6 is equal to 2.

If I just close that and since I have recoded it I will have to generate a variable, otherwise it will be merging on the same variable which might be problematic because the same variable we may use it differently. So, better to generate a new variable with the new code. So you just generate as location, simple location which we, or you can simply type loc bracket close. So recode only allows numeric variables.

So this is already in, we will guide it, this data is in string format. I have already told you earlier that the string data does not read the numeric operation. You need to destring it or we will open. We have already destringed it. I will tell how, what do we mean by destringing. And now we have again retain the command recode with the same variables, the same operation, if you enter it gives the result.

You just look at and mark the changes carefully. Loc is not defined with the recode. It is interesting to note that when you have recoded as per our NSS data 1 stands for within household operation, 2 stands for outside, isn't it interesting to label those variables. So you have to label it. We have other slides to be discussed that how to label, 1 stands for within and 2 stands for outside. We are going to tell you in a short while. But I will finish the lecture with recoding today. From the next class we will discuss, but let me define the value labeling from the next



class. Egen then we have other things, but let me stop with this recoding till this and you may practice in between and raise your doubts if any. We will continue from the next class.

With this, thank you all.