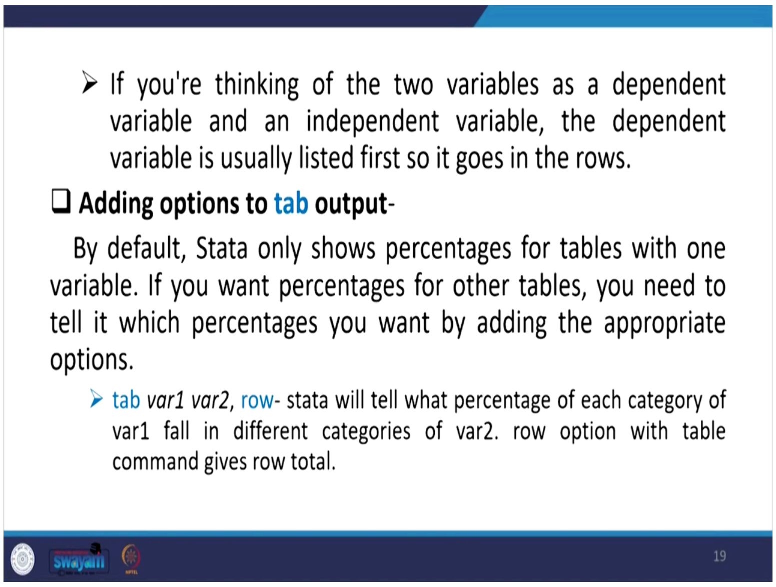
Exploring Survey Data on Health Care

Prof. Pratap C. Mohanty
Department of Humanities and Social Sciences
Indian Institute of Technology, Roorkee

Lecture - 20
Data Browse and Basic Statistics - II

Welcome friends once again to the NPTEL MOOC module on handling Healthcare Data. We are on the 4th week and about to finish the 4th week and this particular lecture is in addition to the lecture; we have taken on the previous class that is Data Browse and Basic Statistics.

(Refer Slide Time: 00:40)



- If you're thinking of the two variables as a dependent variable and an independent variable, the dependent variable is usually listed first so it goes in the rows.

Adding options to tab output-

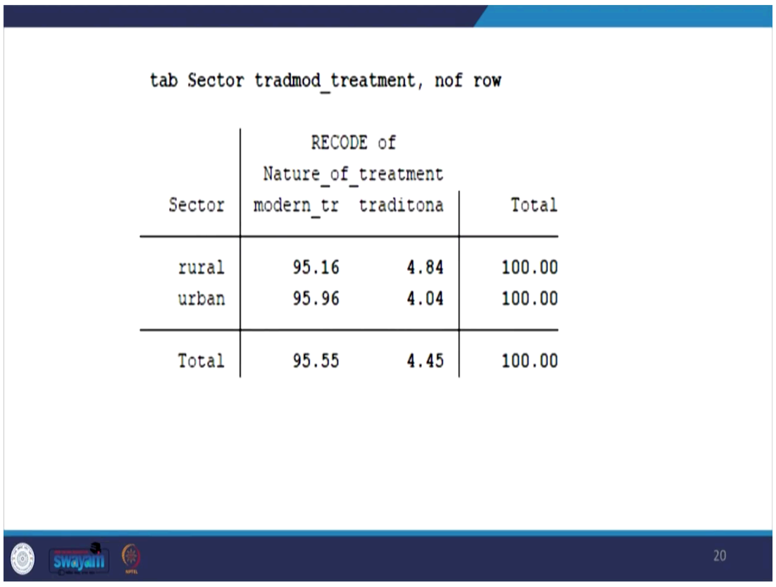
By default, Stata only shows percentages for tables with one variable. If you want percentages for other tables, you need to tell it which percentages you want by adding the appropriate options.

- tab *var1 var2*, row- stata will tell what percentage of each category of var1 fall in different categories of var2. row option with table command gives row total.

If you are thinking of two variables as a dependent variable and an independent variable, the dependent variable is usually listed first. So, it goes in the row. So, usually when we do some sort of cross tabulations or deriving this kind of two way table results, we should keep in our mind that which one is going to be a dependent variable and which one is going to be independent variable and so, that you will have more meaningful interpretations.

We need to add options like I have already mentioned, we can give row options or column options and its results how it appears, how it is useful; we have already explained.

(Refer Slide Time: 01:29)

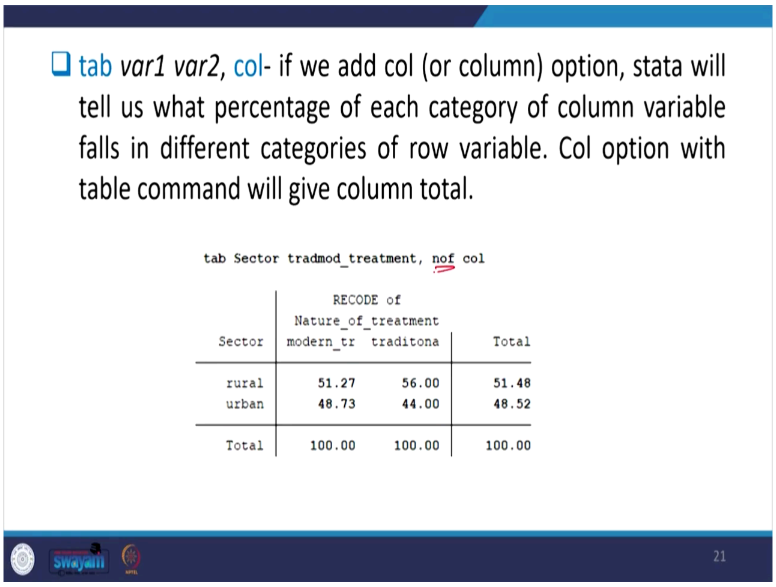


```
tab Sector tradmod_treatment, nof row

           |       RECODE of
           |   Nature_of_treatment
    Sector | modern_tr  traditona |     Total
-----------+----------------------+----------
     rural |     95.16       4.84 |    100.00
     urban |     95.96       4.04 |    100.00
-----------+----------------------+----------
     Total |     95.55       4.45 |    100.00
```

20

So, if we even *nof* command if you give, it tab two variable, this is first variable, this is second variable. If you *nof* and *row*; so, it is only going to give you the percentages.

(Refer Slide Time: 01:50)

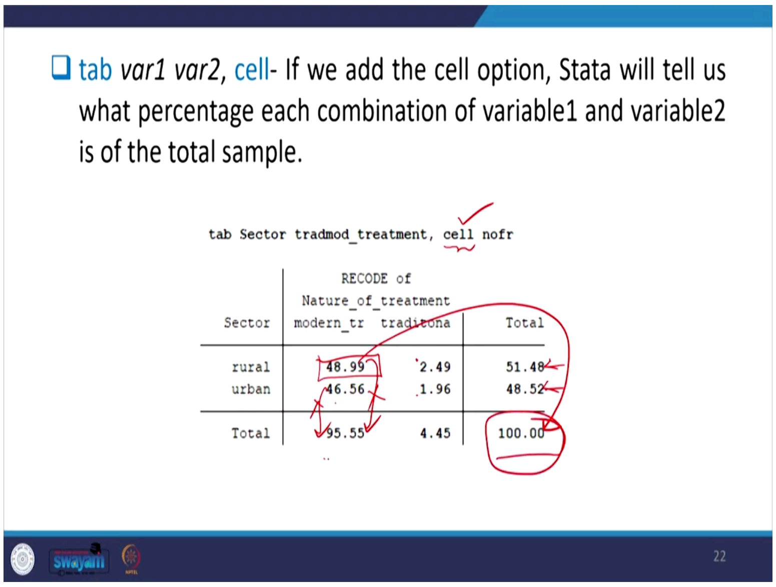


- tab *var1 var2*, col- if we add col (or column) option, stata will tell us what percentage of each category of column variable falls in different categories of row variable. Col option with table command will give column total.

```
tab Sector tradmod_treatment, nof col

           |       RECODE of
           |   Nature_of_treatment
    Sector | modern_tr  traditona |     Total
-----------+----------------------+----------
     rural |     51.27      56.00 |     51.48
     urban |     48.73      44.00 |     48.52
-----------+----------------------+----------
     Total |    100.00     100.00 |    100.00
```

21

(Refer Slide Time: 01:58)



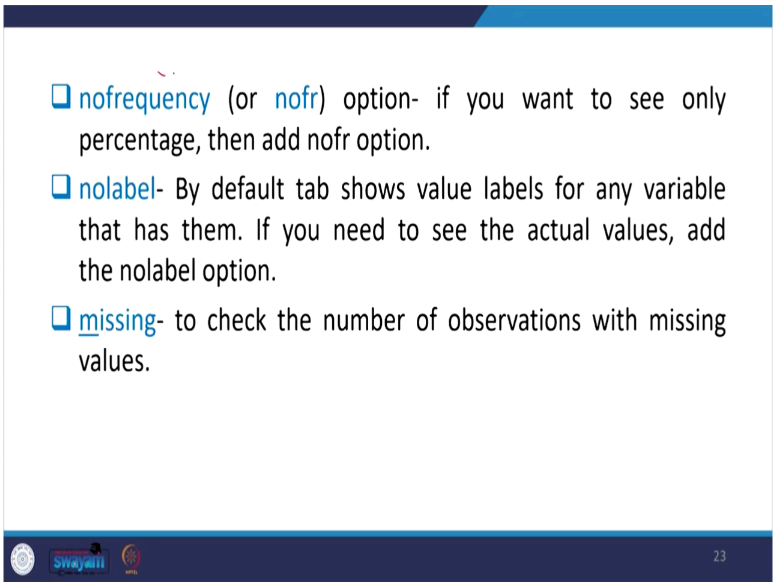
- tab *var1 var2*, cell- If we add the cell option, Stata will tell us what percentage each combination of variable1 and variable2 is of the total sample.

```
tab Sector tradmod_treatment, cell nofr

           |      RECODE of
           | Nature_of_treatment
    Sector | modern_tr  traditiona |     Total
-----------+----------------------+----------
     rural |     48.99       2.49 |     51.48
     urban |     46.56       1.96 |     48.52
-----------+----------------------+----------
     Total |     95.55       4.45 |    100.00
```

Similarly, *nof column* you can do it, you can add extra option that is going to give you column wise percentages. Most important though less often used, but very rarely discussed, you can also take a note called cell percentage. Like we compare these out of these or these out of these out of these or these out of these, if it is column percentages or if it is by row percentage we compare with the row. But, in this case this is not going to be derived, this is going to be straightway derived from the 100 of this.

So, out of total persons both rural urban and in this case nature of treatment in our discussion, how much percentage carry on one cell. This is one cell, this is second cell, this is third cell, four cell. In each cell what is the percentages, this is also another cell, this is another cell. So, out of these would be your 100 instead of either row or in the column.

(Refer Slide Time: 03:01)



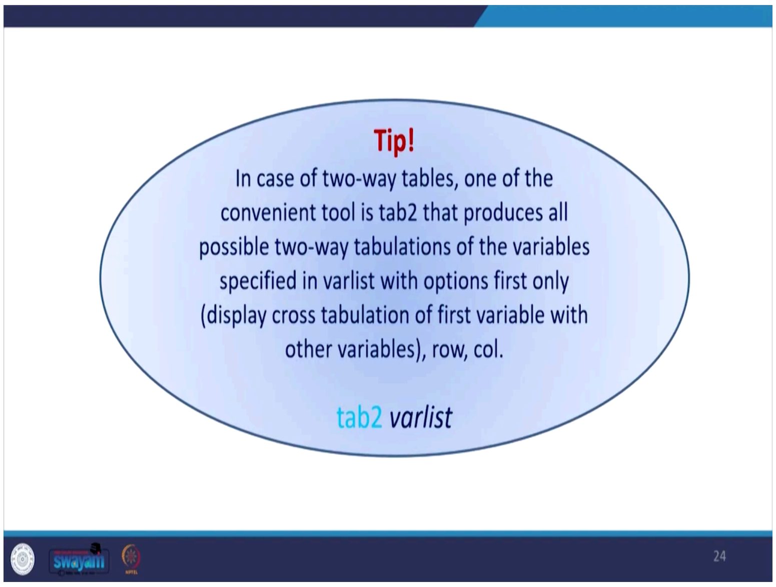
- nofrequency (or nofr) option- if you want to see only percentage, then add nofr option.
- nolabel- By default tab shows value labels for any variable that has them. If you need to see the actual values, add the nolabel option.
- missing- to check the number of observations with missing values.

23

So, no frequency command and it is details we explain already through operation. Then second one is called nolabel by default tab shows value levels for any variable, that has already entered with them. If you need to see the actual values are the nolabel option, if you write nolabel then the actual code name will be display on the result.

Since we have not labelled in our variable, so, we are not going to run this command at this moment, you can label, I will discuss about labelling, then we can run it accordingly. Missing: to check the number of observation with missing values, missing command could also be added as an option alright, that will be give you or will be giving you missing values.

(Refer Slide Time: 03:55)



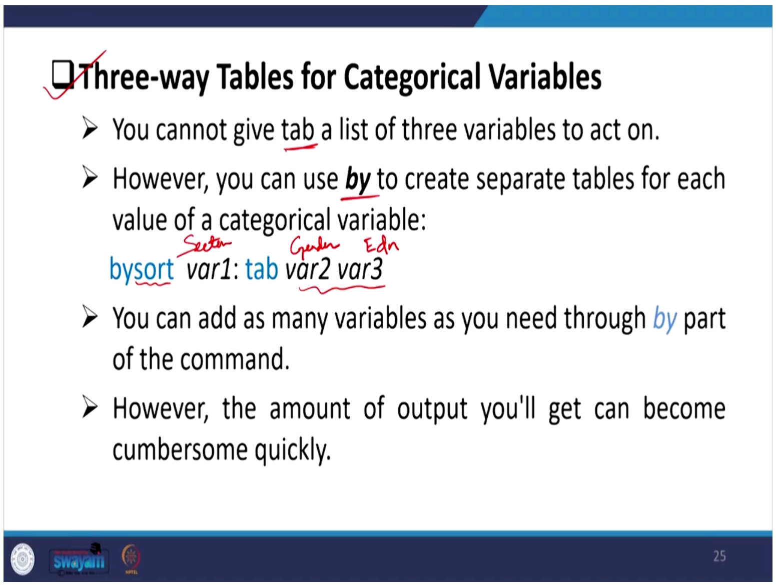
Tip!

In case of two-way tables, one of the convenient tool is tab2 that produces all possible two-way tabulations of the variables specified in varlist with options first only (display cross tabulation of first variable with other variables), row, col.

tab2 *varlist*

In case of two-way tables, one of the convenient tool is tab2 that produces all possible two way tabulations of the variables specified in variable list with options first only, that is displayed cross tabulation of first variable with other variables row and column wise. So, tab2 and variable, all the operation you can do it.

(Refer Slide Time: 04:25)



❑ 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 $\overset{\text{Sector}}{var1}$: tab $\overset{\text{Gender}}{var2}$ $\overset{\text{Edn}}{var3}$
- 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.

You can just operate on your own and I am sure that is going to be helpful. Another interesting table for us to operate is called three-way, table three way table is often required for the researcher who wishes to derive results not just by two variables, that might be third interaction, third common point through the third variable.

So, like for example, we explained here sector and gender like the third variable could be their education. How education is actually linked to other two variables or sector how maybe your first variable and other two would be gender and education. So, you cannot give tab list and all those three variables to give you the result, just tab and on all those list of three variables is going to through you error.

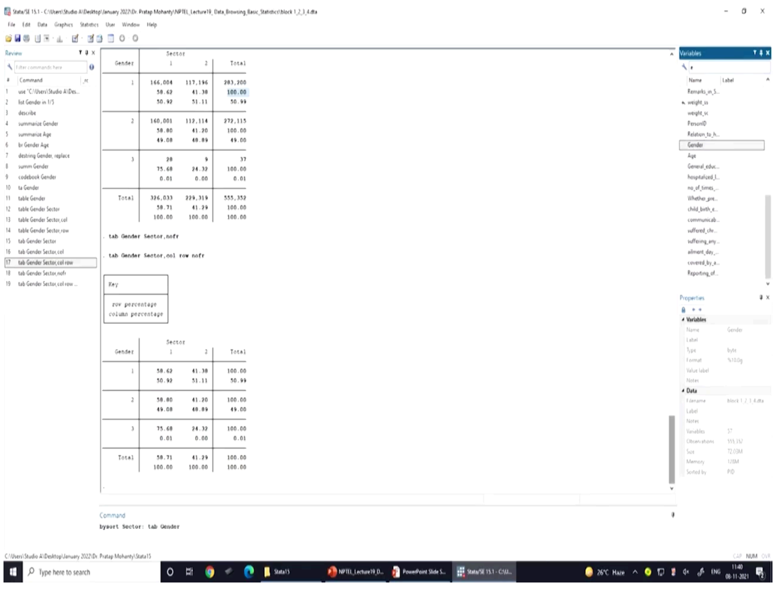
However, you can apply one command called *bysort* command, *bysort* command sort command because it will sort the first variable. First variable that is, suppose you a enter rural here first or sector, if it is sector and this is your gender and this is your education. So, sector it will actually sort the variable and it is entry, then it will compare the tabulation of these two other variables.

You can add as many variables as you need through by part of the command if. So, by part of the command we can actually add more variables as well.

However, the amount of output you will get can become cumbersome quickly. If it is instead of 2 layers or 3 layers, if there are 4 layers of information and there are intricacies explained, it is very difficult to read at a go. So, that is why the results are going to be very complicated.

So, usually we operate through at maximum 3, 3 through this command and we can also operate here bysort, then there will be bysort command will be taking three variables here.

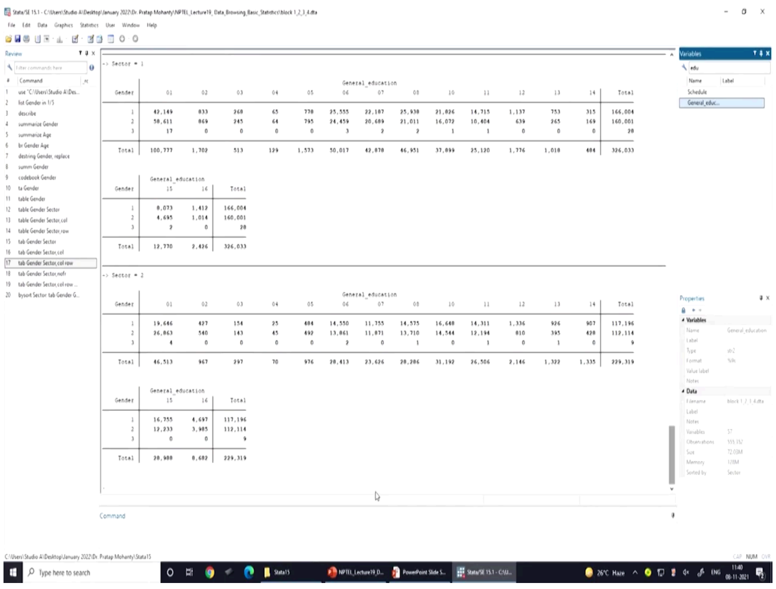
(Refer Slide Time: 07:03)



```
             |        Sector
      Gender |         1          2 |     Total
-------------+----------------------+----------
           1 |   166,004    117,196 |   283,200
             |     58.62      41.38 |    100.00
             |     50.92      51.11 |     50.99
-------------+----------------------+----------
           2 |   160,001    112,114 |   272,115
             |     58.80      41.20 |    100.00
             |     49.08      48.89 |     49.00
-------------+----------------------+----------
           3 |        28          9 |        37
             |     75.68      24.32 |    100.00
             |      0.01       0.00 |      0.01
-------------+----------------------+----------
       Total |   326,033    229,319 |   555,352
             |     58.71      41.29 |    100.00
             |    100.00     100.00 |    100.00

. tab Gender Sector,nofr

. tab Gender Sector,col row nofr

+-------------------+
| Key               |
|-------------------|
|  row percentage   |
| column percentage |
+-------------------+

             |        Sector
      Gender |         1          2 |     Total
-------------+----------------------+----------
           1 |     58.62      41.38 |    100.00
             |     50.92      51.11 |     50.99
-------------+----------------------+----------
           2 |     58.80      41.20 |    100.00
             |     49.08      48.89 |     49.00
-------------+----------------------+----------
           3 |     75.68      24.32 |    100.00
             |      0.01       0.00 |      0.01
-------------+----------------------+----------
       Total |     58.71      41.29 |    100.00
             |    100.00     100.00 |    100.00

.

Command
bysort Sector: tab Gender
```

Let be sector is your first variable bysort sector; then colon then our tab, tab2 variables let be first variable is our gender. Then another categorical variable we can find out from the list of the sample data, that is being shared to you; general education. So, this is our command alright.

(Refer Slide Time: 07:45)



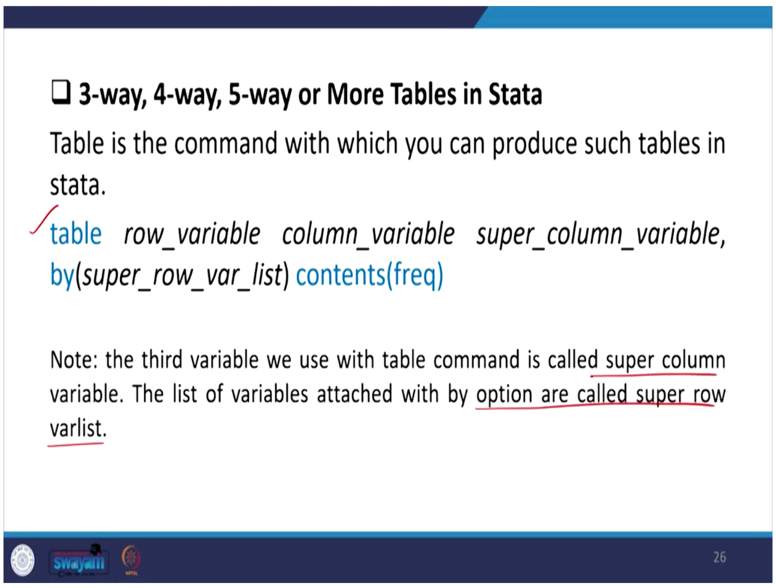
```
-> Sector = 1

             |                                     General_education
      Gender |        01         02         03         04         05         06         07         08         10         11         12         13         14 |     Total
-------------+-------------------------------------------------------------------------------------------------------------------------------------------------+----------
           1 |    42,149        833        268         65        778     25,555     22,187     25,938     21,026     14,715      1,137        753        315 |   166,004
           2 |    58,611        869        245         64        795     24,459     20,689     21,011     16,072     10,404        639        265        169 |   160,001
           3 |        17          0          0          0          0          3          2          2          1          1          0          0          0 |        28
-------------+-------------------------------------------------------------------------------------------------------------------------------------------------+----------
       Total |   100,777      1,702        513        129      1,573     50,017     42,878     46,951     37,099     25,120      1,776      1,018        484 |   326,033

             | General_education
      Gender |        15         16 |     Total
-------------+----------------------+----------
           1 |     8,073      1,412 |   166,004
           2 |     4,695      1,014 |   160,001
           3 |         2          0 |        28
-------------+----------------------+----------
       Total |    12,770      2,426 |   326,033

-> Sector = 2

             |                                     General_education
      Gender |        01         02         03         04         05         06         07         08         10         11         12         13         14 |     Total
-------------+-------------------------------------------------------------------------------------------------------------------------------------------------+----------
           1 |    19,646        427        154         25        484     14,550     11,755     14,575     16,648     14,311      1,336        926        907 |   117,196
           2 |    26,863        540        143         45        492     13,861     11,871     13,710     14,544     12,194        810        395        428 |   112,114
           3 |         4          0          0          0          0          2          0          1          0          1          0          1          0 |         9
-------------+-------------------------------------------------------------------------------------------------------------------------------------------------+----------
       Total |    46,513        967        297         70        976     28,413     23,626     28,286     31,192     26,506      2,146      1,322      1,335 |   229,319

             | General_education
      Gender |        15         16 |     Total
-------------+----------------------+----------
           1 |    16,755      4,697 |   117,196
           2 |    12,233      3,985 |   112,114
           3 |         0          0 |         9
-------------+----------------------+----------
       Total |    28,988      8,682 |   229,319

.

Command
```

So, here you will get all sort of information. Why bysort? First is your sector bysort it is sorted out sector two with the first entry is 1, then the second result will be on 2, 2 is or maybe urban area, first is rural area, gender and education. So, gender we have entered first.

So, it has taken on the row and education on our second variable on the tab command, that is displayed on the column variable.

So, like here in rural area what is the interpretation? In the rural area, how many males are having in 1 level of education or how many males are having 16th years of education of general education. So, this is possible to find out the third information as well. So, on board in the two-way table we could not able to get the third information, here it has given better idea for interpretation. So, this is what we have already guided.

(Refer Slide Time: 09:00)



❑ **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)

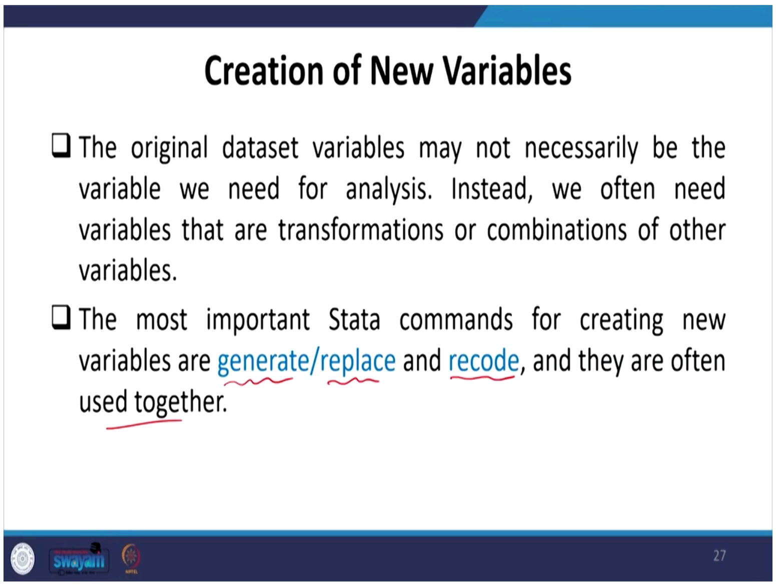
Note: the third variable we use with table command is called super column variable. The list of variables attached with by option are called super row varlist.

26

Similarly 3 way, 4 way, 5 way and more tables in Stata is possible, instead of tab command if you give simply table command, table command is effective more when you have more than two, three variables. Table commands what it takes? Like your row variable then column variable, then your super column variable then you can actually sort it by your super row variable list. So, then you can add some content like frequencies, it will give you the result as frequency.

The note here we have mentioned for you is that the third variable, we use with table command is called super column variable. Basically, after two variable then with third layer called that variable is called a super column variable. The list of variables attached with by options are called super row variables. So, by option once we have sorted out through the row then we can find out this very clearly.

(Refer Slide Time: 10:13)



Creation of New Variables

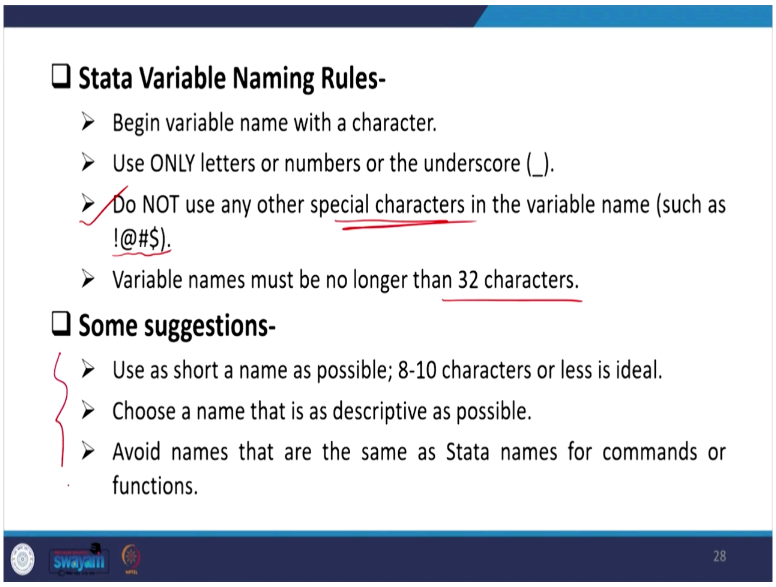
- ❑ The original dataset variables may not necessarily be the variable we need for analysis. Instead, we often need variables that are transformations or combinations of other variables.
- ❑ The most important Stata commands for creating new variables are generate/replace and recode, and they are often used together.

27

Creation of new variables, how to have new variables, we can create in Stata. The original data set variables may not necessarily be the variable we need for analysis. Instead, we often need variables that are transformations or combinations of other variables. The most important data commands for creating new variable are like generate replace or recode.

These three we are going to explain you ,what are the differences in each case? They are often used together, but there are certain differences. Even those who are familiar with Stata, you might be confused with these three, let us get understood very clearly.

(Refer Slide Time: 10:57)



❑ 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.

❑ Some suggestions-

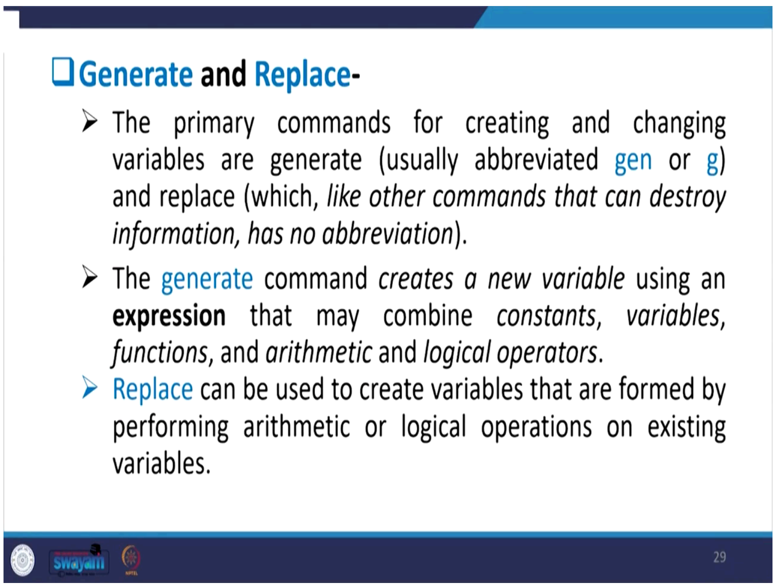
- ➢ Use as short a name as possible; 8-10 characters or less is ideal.
- ➢ Choose a name that is as descriptive as possible.
- ➢ Avoid names that are the same as Stata names for commands or functions.

28

Stata variable naming rules, how we can name the Stata variables. Begin variable name with a character, then use ONLY letters and numbers or the underscore. Then do NOT use that is more important, do not use these letters, these special characters are not allowed; special characters like these are not allowed while naming a variable in Stata. Variable names must be no longer than 32 characters. So, more than 32 characters Stata is not going to name it or consider as a name.

So, some suggestions we have mentioned here like use as sort a name as possible; 8 to 10 characters are usually considered to be ideal name. Choose a name that is as descriptive as possible. Avoid names that are the same as Stata names for commands for functions. Like function command and the variable command, if you just simply give the same, it might be makes you more confused; so, better to avoid all those things. This is quite important, those are very new and at the entry level.

(Refer Slide Time: 12:24)



❑ 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.

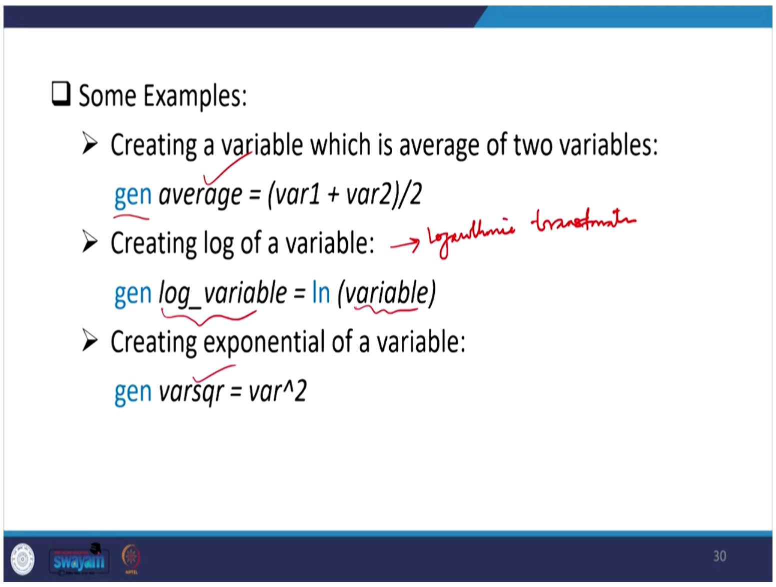
29

So, we have to clarify these two again; generate and replace. Then what do you mean by generate first? The primary commands for creating and changing variables are called generate. They are usually abbreviated with gen or g and replace like with other command, that can destroy information and has no abbreviation. So, replace word has to be carried as it is.

The generate command creates a new variable use certain expression. So, you have to give like if you are generating a variable, then you have to express it with certain expression.

Expression might be with some arithmetic equation, function or logical operation operators. Replace can be used to create variables that are formed by forming arithmetic or logical operations on existing variables.

(Refer Slide Time: 13:28)



- Some Examples:
  - Creating a variable which is average of two variables:

    gen average = (var1 + var2)/2
  - Creating log of a variable: $\rightarrow$ logarithmic transformation

    gen log_variable = ln (variable)
  - Creating exponential of a variable:

    gen varsqr = var^2

30

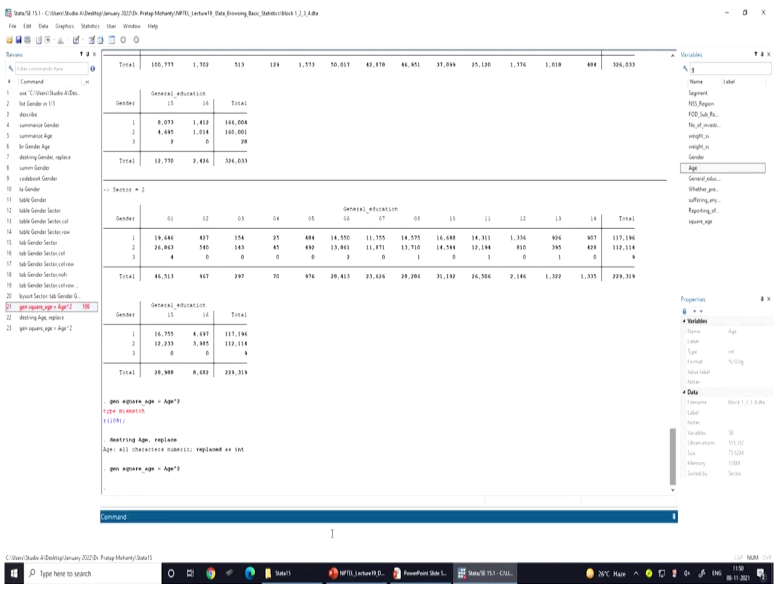
Some examples where I put it; you can experiment on your own because, of time paucity for the class, I am not operating everything. But, the same approach you can apply and find out, like generate (gen) you have to write it down. Then if you want two variables to be averaged, two numeric values you want, this would be averaged by 2 or if three variables are by 3 then you have to divide it by 3.

So, that will give you average value with this gen command. Then creating a log of a variable like log of variable means here, we are saying logarithmic transformation. So, here by default it considered natural log. So, generate log variable we have given this name as log underscore name of the variable maybe expenditure. So, log expenditure log underscore expenditure, then within bracket the variable name maybe initial, it is expenditure.

Then log underscore expenditure is equal to ln within bracket expenditure variable name is expenditure, that will be generating a new variable log variable. So, like here squaring a variable another one like we are generating a variable we need to square it. Why squaring? Like some variables by default considered to be linear and the linearity of the variable is not going to give much difference in the model.

So, if you want that some sort of major jumps or some non-linearity in the data is required for identifying the role of that particular variable, then you can take the square of that variable or even sometimes cube of that variable. How we can do it? Like in your data, I am just going to show it, it is on the data one age variable is with us, we usually do that.

(Refer Slide Time: 15:46)



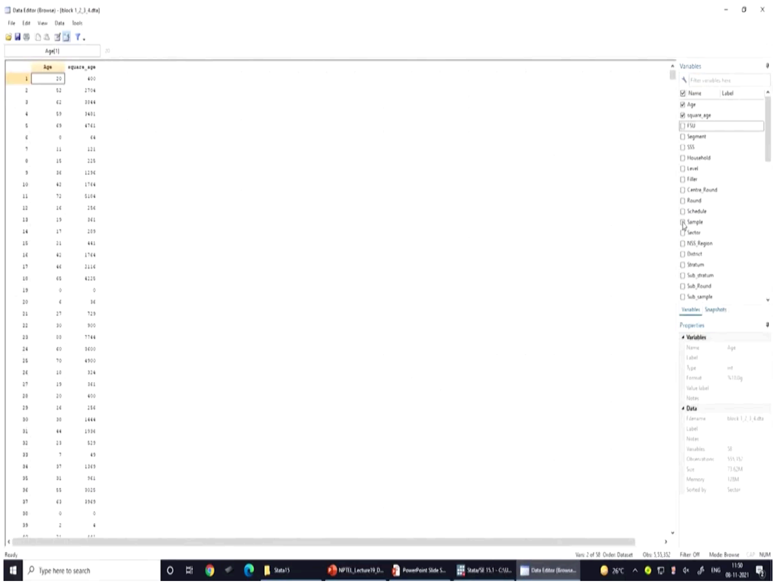
```
    Total |   100,777      1,702        513        129      1,573     50,017     42,070     46,951     37,099     25,120      1,776      1,010        404 |   326,033

          |   General_education
   Gender |        15         16 |     Total
----------+----------------------+----------
        1 |     8,073      1,412 |   166,004
        2 |     4,695      1,014 |   160,001
        3 |         2          0 |        28
----------+----------------------+----------
    Total |    12,770      2,426 |   326,033

-> Sector = 2

          |                                                          General_education
   Gender |        01         02         03         04         05         06         07         08         10         11         12         13         14 |     Total
----------+----------------------------------------------------------------------------------------------------------------------------------------------------+----------
        1 |    19,646        427        154         25        404     14,550     11,755     14,575     16,640     14,311      1,336        926        907 |   117,196
        2 |    26,863        540        143         45        492     13,061     11,071     13,710     14,544     12,194        810        395        420 |   112,114
        3 |         4          0          0          0          0          2          0          1          0          1          0          1          0 |         9
----------+----------------------------------------------------------------------------------------------------------------------------------------------------+----------
    Total |    46,513        967        297         70        976     28,413     23,626     28,206     31,192     26,506      2,146      1,322      1,335 |   229,319

          |   General_education
   Gender |        15         16 |     Total
----------+----------------------+----------
        1 |    16,755      4,697 |   117,196
        2 |    12,233      3,985 |   112,114
        3 |         0          0 |         9
----------+----------------------+----------
    Total |    28,988      8,682 |   229,319

. gen square_age = Age^2
type mismatch
r(109);

. destring Age, replace
Age: all characters numeric; replaced as int

. gen square_age = Age^2
```

So, let us see how age variable is whether we have squared it or not we need to just check first. So, we have taken age here and we will use define a new variable called gen; then variable name is square age maybe square age underscore age. So, is equal to age to the power 2. So, power will be using this n 2.

So, this will be converting the variable again, there are some problem because of the data by default is in strings, we have to convert these two, destring first, we will make it to numeric value; destring age then we will go by the same command and replace; so it replace the original variable, replace command you can just take a note how why we applied a replace. Replace because, we have destring change the nature of the variable and now if we do not replace, then there will be some mismatch.

So, now our variable is in numeric entry, numeric values. Now, if you apply the command generate square underscore age, we have defined a new variable name, then equal to age to the power 2, then this is going to convert it. Now, on the variable list you can see a new variable has already been generated, that is with the name square underscore age. Now, you can check how whether it has been squared or not br of these two variables together, age and square age, *br age age^2*. Now, you can just have a check.

(Refer Slide Time: 17:48)



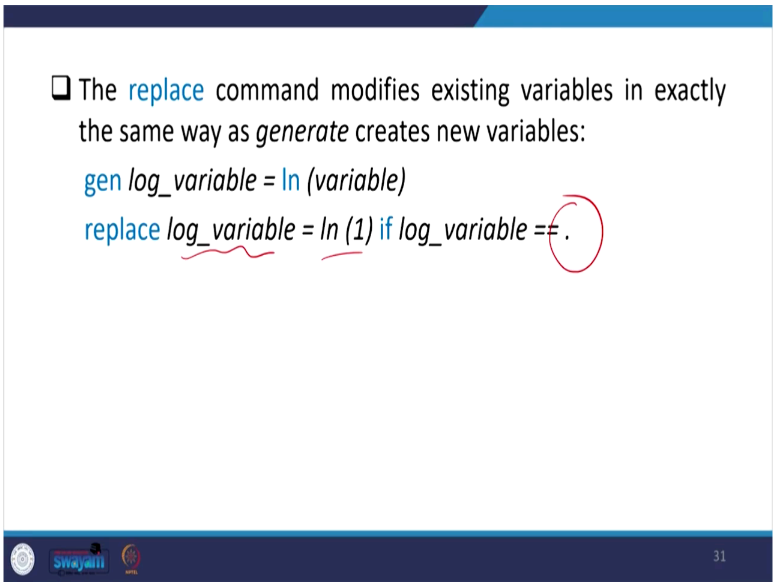
Data Editor (Browse) - [block 1_2_3_4.dta]

Age[1] 20

	Age	square_age
1	20	400
2	52	2704
3	62	3844
4	59	3481
5	69	4761
6	8	64
7	11	121
8	15	225
9	36	1296
10	42	1764
11	72	5184
12	16	256
13	19	361
14	17	289
15	21	441
16	42	1764
17	46	2116
18	65	4225
19	0	0
20	6	36
21	27	729
22	30	900
23	88	7744
24	60	3600
25	70	4900
26	18	324
27	19	361
28	20	400
29	16	256
30	38	1444
31	44	1936
32	23	529
33	7	49
34	37	1369
35	31	961
36	55	3025
37	63	3969
38	0	0
39	2	4

Now, just have a comparison age 20, now it has been square as 400, 52, then its square is 2704 and etc, 6, 36 etc. all have been square.

(Refer Slide Time: 18:06)



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

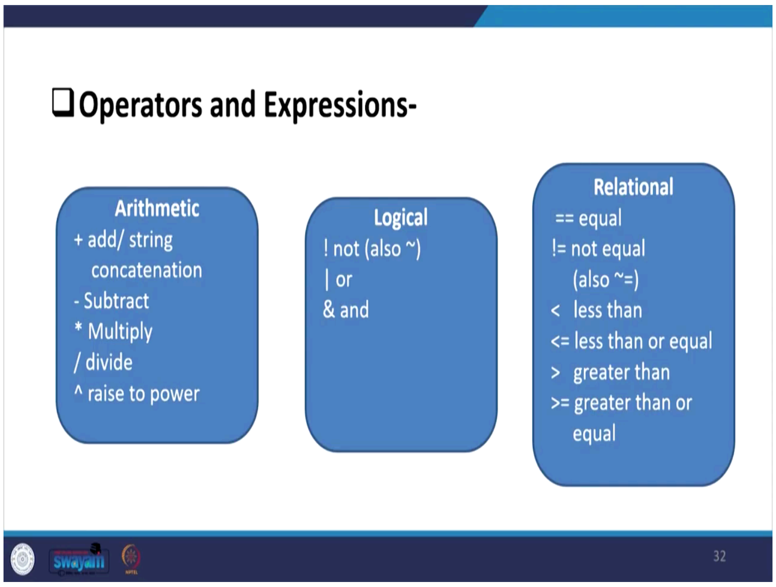
  gen *log_variable* = ln *(variable)*

  replace *log_variable* = *ln (1)* if *log_variable* == .

So, let us make a move to our next explanation. Replace we have already shown through the operation. The replace command modifies existing variables in exactly the same way is and it creates new variables. So, generate log if logarithmic transformation will do it, then you can also replace the variable if there are some missing values in the data entry, there are some missing values as dot; that you can replace, replace to 1 log 1.

So, in replace you have mentioned the log variable; so, missing values are there. So, what we do, we have change is to log 1. So, that can able to replace your value.

(Refer Slide Time: 19:00)



❑Operators and Expressions-

Arithmetic
+ add/ string concatenation
- Subtract
* Multiply
/ divide
^ raise to power

Logical
! not (also ~)
| or
& and

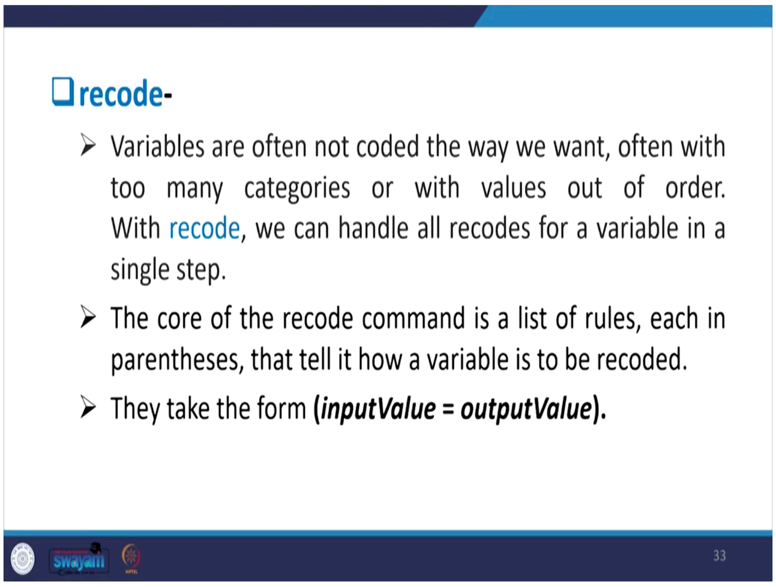
Relational
== equal
!= not equal (also ~=)
< less than
<= less than or equal
> greater than
>= greater than or equal

32

There are certain operators and expressions operators which we add in addition, options we give in addition to our main command. Like arithmetic operation we add like subtract, multiply or divide or raise to the power. So, their symbol will apply, adding will simply give this kind of command, plus, a subtract etc.

So, some logical command like if it is not, this is not equal to this; then this command has to be given and is for this and signs that could be given. Similarly, some relational functions are also given. If you want to define within a range for your variable then command less than or equal can also be given.

(Refer Slide Time: 19:55)



❑ recode-

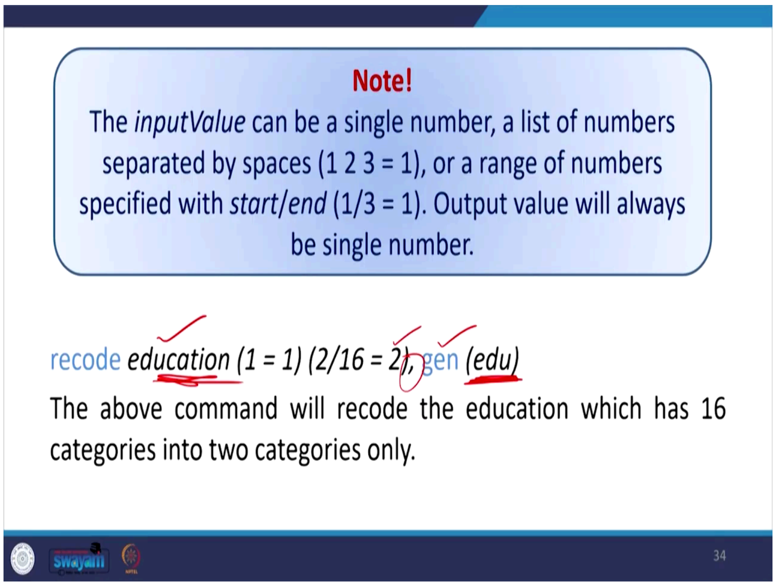
- Variables are often not coded the way we want, often with too many categories or with values out of order. With recode, we can handle all recodes for a variable in a single step.
- The core of the recode command is a list of rules, each in parentheses, that tell it how a variable is to be recoded.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

33

Then, next one is called recode. Recode variables are often not coded the way we want, like any purchase you will get some files called recode, individual recode. They have recoded the file and made handy for your use. 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 parenthesis that tell it how a variable is to be recoded. They can take the form like in input Value is equal to the outputValue, like if recode these to that like; if you simply said recode this equal to that, that will be converting to any variable. And, this is as simple as that, simply recode one variable and its new name if you wanted to give it, it will recode it accordingly.

(Refer Slide Time: 21:29)



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 education (1 = 1) (2/16 = 2), gen (edu)
```

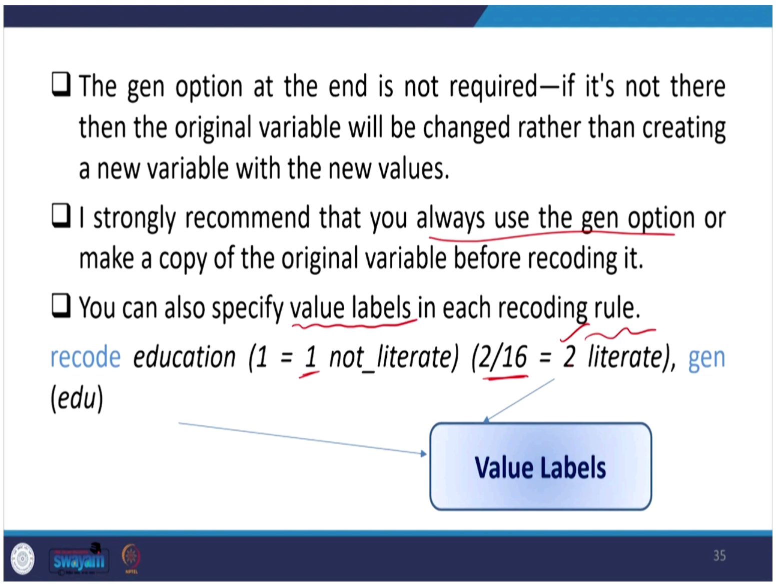
The above command will recode the education which has 16 categories into two categories only.

So, then the inputValue can be a single number, the list of numbers separated by spaces like 1 2 3, if want to recode like their entries are 1 2 3, you wanted to make all those 1 2 2 entries to 1. And maybe 4, 5, 6 to 2 or maybe 7, 8, 9, 10 and onwards everything to be 4, like standard years of education. You want to keep it till 5 years age as a primary, upper primary from 6 to 10 suppose you wanted to keep it.

So, then from 1 till 5 you wanted to keep a code as 1. So, that is going to recode it with 1. So, output value will always be a single number, a recode like here education I have mentioned as 1 you wanted to keep as 1. So, it will continue with 1, then rest 2 to 16 like years of education till 16 are there, all other you wanted to keep it as 2. Maybe 1 year you wanted to understand or give importance on the only persons is having education or not, then more than 1 years of experience or not.

So, your two commands are specially going to give you whether the person is educated with more than 2 years of education or not. So, then generated a variable, the original variable is this one education, the new variable we have generated it we have given the name within bracket as edu and Stata is very case sensitive. So, we have to specify with their particular name and generate and you can write down with a comma, comma should be given, once you have given comma Stata reads that you have started giving options. The above command will recode the education which has 16 categories into two categories only.

(Refer Slide Time: 23:13)



- The gen option at the end is not required—if it's not there then the original variable will be changed rather than creating a new variable with the new values.
- I strongly recommend that you always use the gen option or make a copy of the original variable before recoding it.
- You can also specify value labels in each recoding rule.

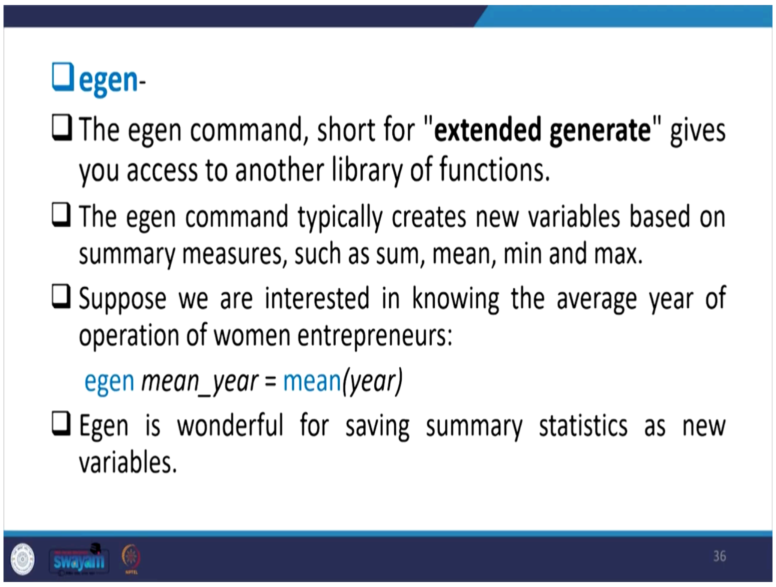
```
recode education (1 = 1 not_literate) (2/16 = 2 literate), gen
(edu)
```

The generate option at the end is not required, if it is not there then the original variable will be changed rather than creating a new variable like; if you do not give generate then the same variable will be actually changed with its recoded value. But, we are suggesting that you should give a new name, you might require the first variable as well. So, it is always suggested that you should give a generate command at the end.

I strongly recommend that you should always use the generate option, this is what is written; make a copy of the original variable before recoding it. You can also specify value labels in each recoding rule like recode education 1 equal to 1 is not_literate. Like the value labels can also be mentioned at the time of recoding, not just the recoding we have recorded 2 to 16 as 2, but now we have defined they are called literate.

Suppose, you were saying they are the literate one; so, 2 will be labelled with a name called label, not level, labelled as literate. So, and at the end we have generated. So, label can also be defined, label of the value can also be defined.

(Refer Slide Time: 24:28)



❑ **egen-**

❑ The egen command, short for "**extended generate**" gives you access to another library of functions.

❑ The egen command typically creates new variables based on summary measures, such as sum, mean, min and max.

❑ Suppose we are interested in knowing the average year of operation of women entrepreneurs:

egen *mean_year* = mean*(year)*

❑ Egen is wonderful for saving summary statistics as new variables.

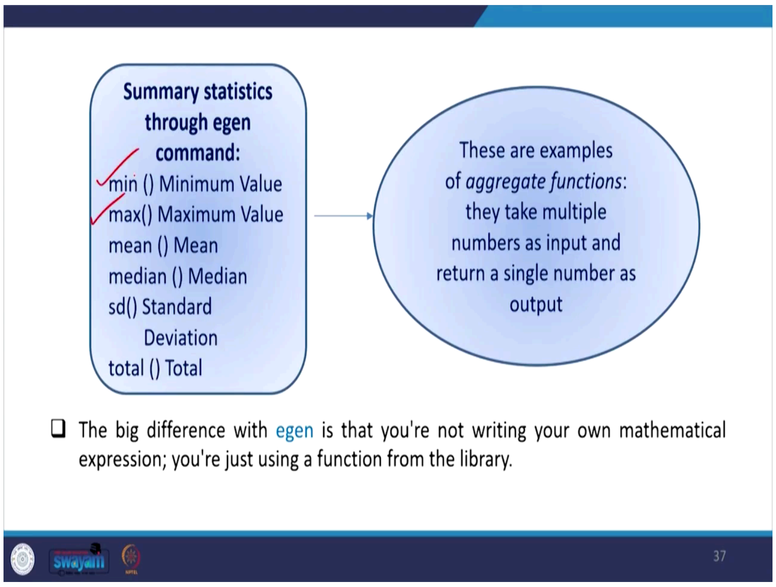
36

So, a couple of more information are there we are going to clarify now; egen is very rarely used, but very important as well, egen command like sort for extended gen. This is in short we write as extended generate format gives you access to another library of functions; egen actually going to give you another library of functions, egen command typically creates new variable based on summary measures such as sum, mean, minimum, maximum etc.

Suppose we are interested in knowing the average year of an operation of women entrepreneurs or even average year of expenditure in health care for the person. So, health care expenditure instead of calculating the average with the function like this number of year now by 12; if we give it that way we can derive it, but here through egen command we can quickly get the mean value.

If year is our variable name, then mean of that year is again written here as the variable name. ‘egen’ command we have added here as mean. So, that will be generating a new variable with the mean of that year. Similarly, mean expenditure if you wanted to get that this is in fact, one of the quickest way of getting the extended commands. Egen is wonderful for saving summary statistics is new variables.

(Refer Slide Time: 26:08)


Summary statistics through egen command:

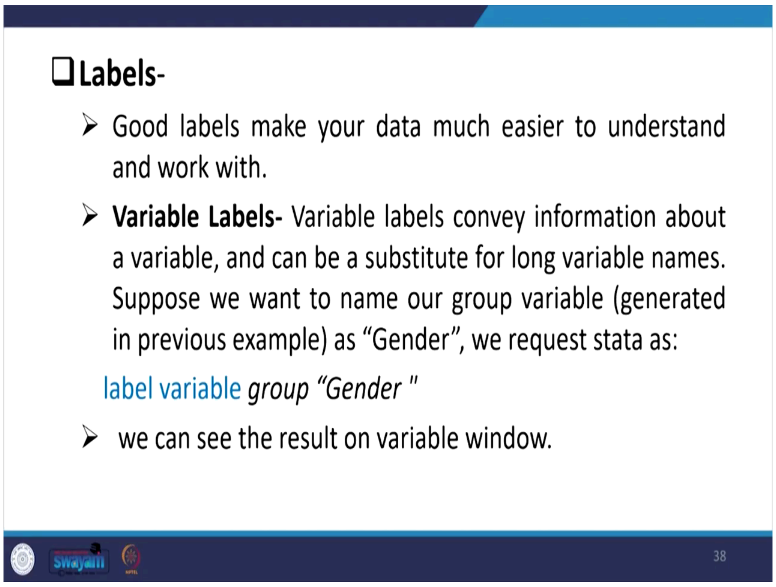
min () Minimum Value
max() Maximum Value
mean () Mean
median () Median
sd() Standard Deviation
total () Total

These are examples of *aggregate functions*: they take multiple numbers as input and return a single number as output

- The big difference with egen is that you're not writing your own mathematical expression; you're just using a function from the library.

So, similarly minimum maximum egen for minimum value maximum value can be defined, I think we have already done it earlier. So, we have shown it how minimum maximum can be defined in our normalization or standardization techniques. So, similarly other indicators you can do it through your egen command. So, rest of the details you can also follow our previous lectures on standardisation, where we have already entered those commands.

(Refer Slide Time: 26:41)

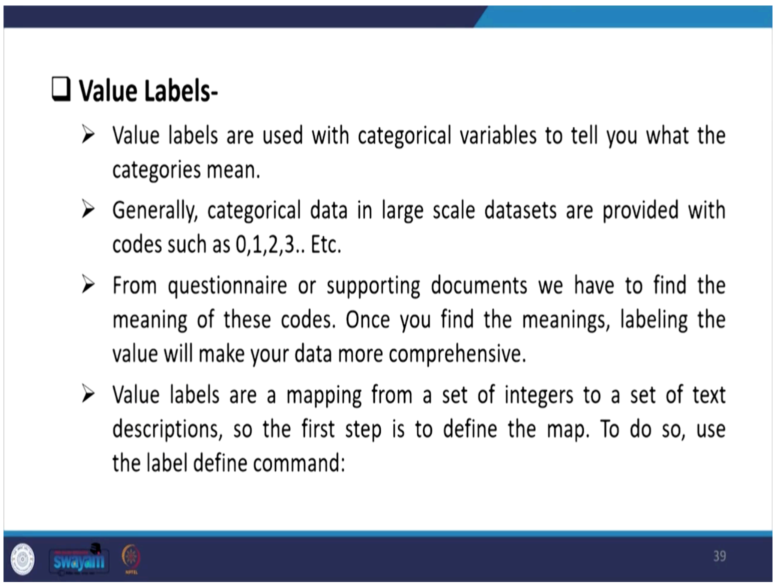

Labels-

- Good labels make your data much easier to understand and work with.
- **Variable Labels-** Variable labels convey information about a variable, and can be a substitute for long variable names. Suppose we want to name our group variable (generated in previous example) as "Gender", we request stata as:

  label variable *group "Gender "*
- we can see the result on variable window.

Last couple of information I am going to clarify on labels. So, good labels make your data much easier to understand and work with. Variable labels - variable labels convey information about a variable and can be substitute for long variable names. Suppose you want to name our group variable a “Gender”, we request Stata as label that is the variable name as group to be in name a “Gender”. So, it will change to “Gender” of that particular variable.

(Refer Slide Time: 27:20)



❑ **Value Labels-**

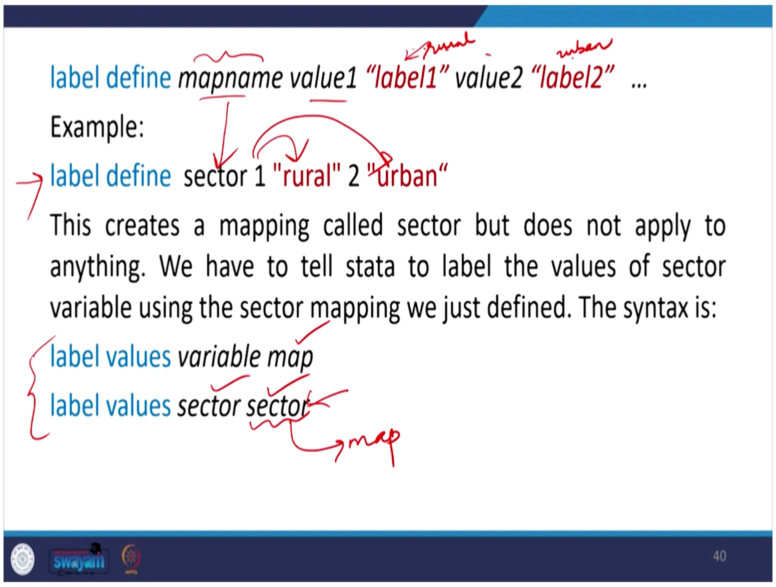
- Value labels are used with categorical variables to tell you what the categories mean.
- Generally, categorical data in large scale datasets are provided with codes such as 0,1,2,3.. Etc.
- From questionnaire or supporting documents we have to find the meaning of these codes. Once you find the meanings, labeling the value will make your data more comprehensive.
- Value labels are a mapping from a set of integers to a set of text descriptions, so the first step is to define the map. To do so, use the label define command:

39

So, we can see those things in our window, and we can operate it, since I have already spent huge time in all those operations, probably I am not going to show it. I am sure that you can do it, if you have still difficulties; I am sure you will come back to us, and we will clarify. Label variable and label values there are differences, value labels are used with categorical variables to tell you what the categories mean.

Like within gender we have values like 0, like 1, 2, 3 etc. So, 1 may stands for a label called gender maybe male, 2 stands for female or the reverse or the maybe not 1 for female etc. But we have to define it from the beginning. From questionnaire or supporting documents we have to find the meaning of these codes. Once you find the meanings labelling the value will make your data more comprehensive. Value labels are a mapping from a set of integers to a set of text descriptions. So, the first step is to define the map.

(Refer Slide Time: 28:33)



label define *mapname value1 "label1" value2 "label2"* ...

Example:

label define sector 1 "rural" 2 "urban"

This creates a mapping called sector but does not apply to anything. We have to tell stata to label the values of sector variable using the sector mapping we just defined. The syntax is:

label values *variable map*

label values *sector sector*

rural
urban
map

So, a map is going to define you map name, let it be value. So, the map name is value1 is like "label1" value2 is "label2", in this like "label2" you can write down as value2 as female, value1 as your male.

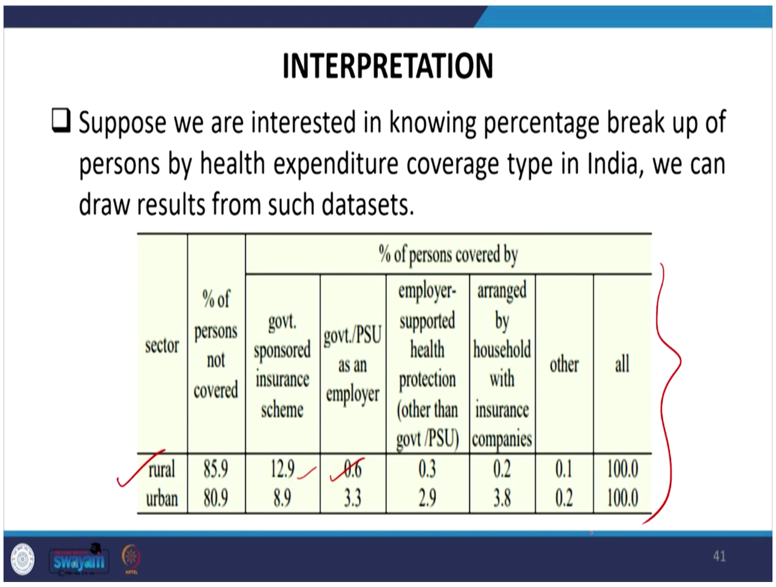
So, map name define mapname, that mapname is called map or a variable name you can say write down code for gender code. If you are just defining this, the first command for us is to go by label define, this is the first command we generally give; label define, what does this mean? Label define we defined in our cases, sector 1 for "rural" and that is map for us is our sector and 1 for "rural" and 2 for "urban".

Now, once the first command is done then second command is label value. What the value we are giving it? Here is label values variable with sector then map, map we have already defined the map. They are gender sector 1 or sector, unlike in case of sector it is rural urban or in case of gender it is male and female.

Now, these two commands should go together; label values, sector and sector, 2 times we have to write it down then it actually sector in this case is our map, this is our map. So, in our map value this is actually rural and this is our urban. So, it will map the codes automatically. Why were developing a map? Because like in your question there might be so many categorical variable with yes, no.

Once you have defined a map name as yes and no, in all variables wherever you wanted to label the values as map with yes, no. So, simply on the place of that you enter that map value mapname. So, it will convert the categorical values with the label with yes and no.

(Refer Slide Time: 30:48)



INTERPRETATION

- Suppose we are interested in knowing percentage break up of persons by health expenditure coverage type in India, we can draw results from such datasets.

sector	% of persons not covered	% of persons covered by					
		govt. sponsored insurance scheme	govt./PSU as an employer	employer-supported health protection (other than govt /PSU)	arranged by household with insurance companies	other	all
rural	85.9	12.9	0.6	0.3	0.2	0.1	100.0
urban	80.9	8.9	3.3	2.9	3.8	0.2	100.0

41

So, these are all details, some interpretation we can do it for our health data, healthcare expenditure coverage that we have been following throughout rural and urban. How many percentage covered in rural, how many percentage covered in urban; whether they are under government insurance scheme or not, whether part of PSUs or not etc.

So, all those operations you can also operate work out through our NSS data and interpret accordingly for your use. This is we have kept it for just for your reference, I hope we have clarified all your details at a go. This lecture I am 100 percent sure going to help you a lot. If you have difficulties, we are happy to clarify it.

Thank you.