Stata Brush Up Course

Session 4

Sergi Quintana

IDEA

September 4, 2024

Plan for the session

We will see the more advanced data options:

- Merge and append.
- Collapse
- Reshape

Append

append appends Stata-format datasets stored on disk to the end of the dataset in memory.

It can be used in different ways:

- With no data in memory:

  append using mydata1 mydata2
- With data in memory:

  append using mydata2

Merge

merge joins corresponding observations from the dataset currently in memory (called the master dataset) with those from filename.dta (called the using dataset), matching on one or more key variables.

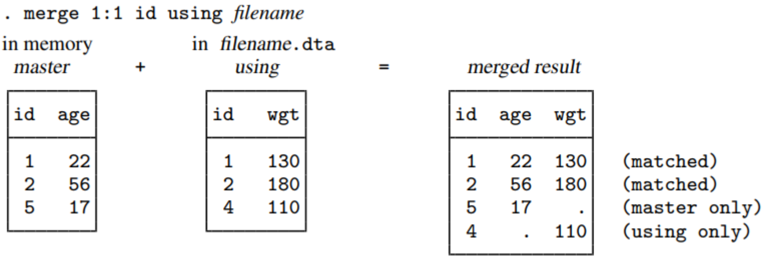
merge can perform to type of merges:

- match merges:
  - one-to-one
  - many-to-one
  - one-to-many
  - many-to-many
- sequential merges

merge is for adding new variables from a second dataset to existing observations. You use merge, for instance, when combining hospital patient and discharge datasets.

Merge

One-to-one merge



```
. merge 1:1 id using filename
```

in memory *master* + in *filename*.dta *using* = *merged result*

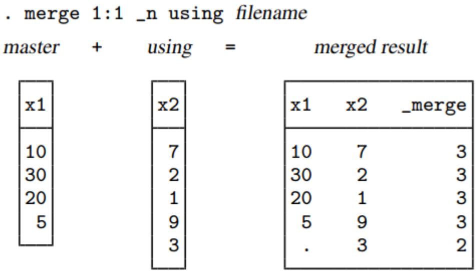
id	age
1	22
2	56
5	17

id	wgt
1	130
2	180
4	110

id	age	wgt	
1	22	130	(matched)
2	56	180	(matched)
5	17	.	(master only)
4	.	110	(using only)

Merge

One-to-one merge



```
. merge 1:1 _n using filename
```

master + *using* = *merged result*

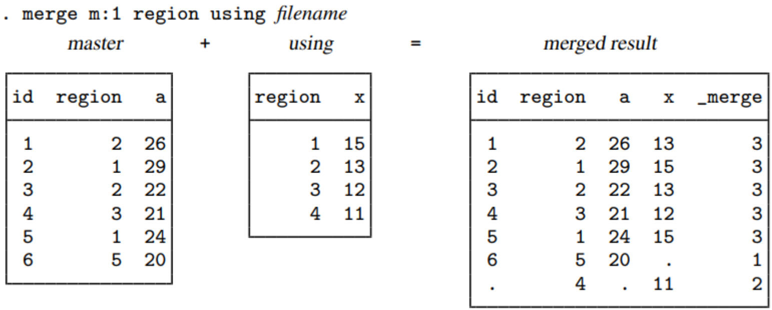
x1
10
30
20
5

x2
7
2
1
9
3

x1	x2	_merge
10	7	3
30	2	3
20	1	3
5	9	3
.	3	2

Merge

Many-to-one merge



```
. merge m:1 region using filename
```

master + *using* = *merged result*

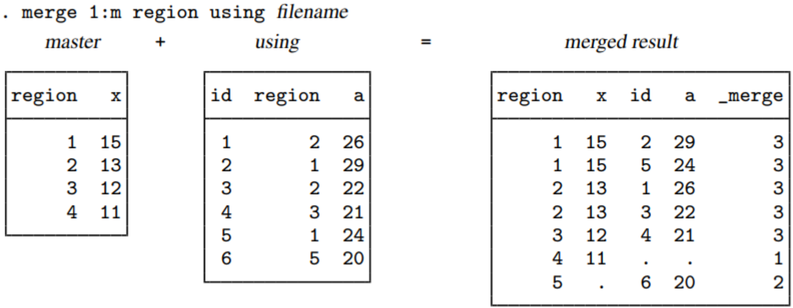
id	region	a
1	2	26
2	1	29
3	2	22
4	3	21
5	1	24
6	5	20

region	x
1	15
2	13
3	12
4	11

id	region	a	x	_merge
1	2	26	13	3
2	1	29	15	3
3	2	22	13	3
4	3	21	12	3
5	1	24	15	3
6	5	20	.	1
.	4	.	11	2

Merge

One-to-many merge



```
. merge 1:m region using filename
```

master + *using* = *merged result*

region	x
1	15
2	13
3	12
4	11

id	region	a
1	2	26
2	1	29
3	2	22
4	3	21
5	1	24
6	5	20

region	x	id	a	_merge
1	15	2	29	3
1	15	5	24	3
2	13	1	26	3
2	13	3	22	3
3	12	4	21	3
4	11	.	.	1
5	.	6	20	2

Merge

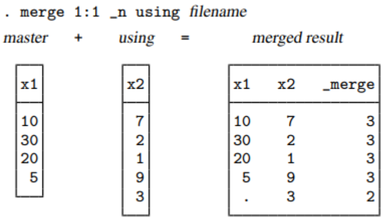
Many-to-many merge

m:m specifies a many-to-many merge and is a bad idea. In an m:m merge, observations are matched within equal values of the key variable(s), with the first observation being matched to the first; the second, to the second; and so on. If the master and using have an unequal number of observations within the group, then the last observation of the shorter group is used repeatedly to match with subsequent observations of the longer group. Thus m:m merges are dependent on the current sort order—something which should never happen.

Because m:m merges are such a bad idea, we are not going to show you an example. If you think that you need an m:m merge, then you probably need to work with your data so that you can use a 1:m or m:1 merge.

Merge

Sequential merge



```
. merge 1:1 _n using filename
```

master + *using* = *merged result*

x1
10
30
20
5

x2
7
2
1
9
3

x1	x2	_merge
10	7	3
30	2	3
20	1	3
5	9	3
.	3	2

Other Methods

There are other methods such as joinby or cross.

- joinby joins, within groups formed by varlist, observations of the dataset in memory with filename, a Stata-format dataset. By join we mean to form all pairwise combinations.

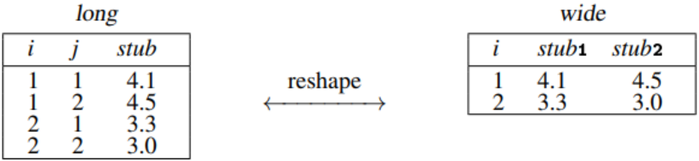
  Observations unique to one or the other dataset are ignored unless unmatched() specifies differently. Whether you load one dataset and join the other or vice versa makes no difference in the number of resulting observations.

- cross forms every pairwise combination of the data in memory with the data in filename. If filename is specified without a suffix, .dta is assumed.

Reshape

reshape converts data from wide to long form and vice versa

It will be a very important tool when dealing with panel data.



long

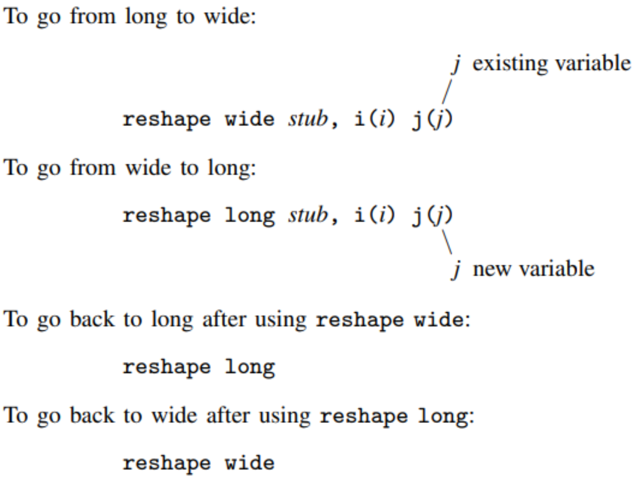
i	*j*	*stub*
1	1	4.1
1	2	4.5
2	1	3.3
2	2	3.0

reshape

wide

i	*stub*1	*stub*2
1	4.1	4.5
2	3.3	3.0

Reshape



To go from long to wide:

```
                                        j existing variable
                                       /
        reshape wide stub, i(i) j(j)
```

To go from wide to long:

```
        reshape long stub, i(i) j(j)
                                   \
                                    j new variable
```

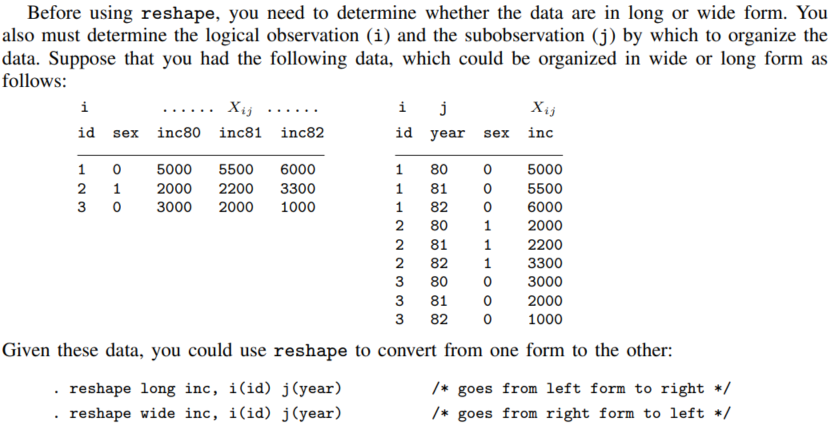
To go back to long after using **reshape wide**:

```
        reshape long
```

To go back to wide after using **reshape long**:

```
        reshape wide
```

Reshape



Before using **reshape**, you need to determine whether the data are in long or wide form. You also must determine the logical observation (i) and the subobservation (j) by which to organize the data. Suppose that you had the following data, which could be organized in wide or long form as follows:

i		......	X_{ij}	......
id	sex	inc80	inc81	inc82
1	0	5000	5500	6000
2	1	2000	2200	3300
3	0	3000	2000	1000

i	j		X_{ij}
id	year	sex	inc
1	80	0	5000
1	81	0	5500
1	82	0	6000
2	80	1	2000
2	81	1	2200
2	82	1	3300
3	80	0	3000
3	81	0	2000
3	82	0	1000

Given these data, you could use **reshape** to convert from one form to the other:

```
. reshape long inc, i(id) j(year)         /* goes from left form to right */
. reshape wide inc, i(id) j(year)         /* goes from right form to left */
```

Exercice - Append

Using the auto data :

- Generate three different 20% random samples and save them as sample1, sample2 and sample3 respectively. You might want to take a look at the command sample: help sample.
- Append the data sets together using the append function.

Exercice - Merge and Reshape

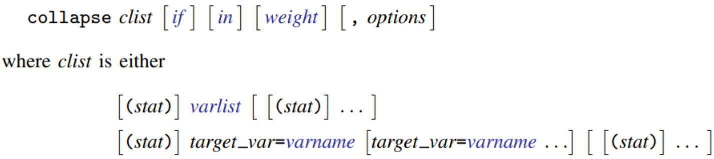
Using the "abdata1":

- Merge it with "abdata2". Make sure you use the right merge. Save it as "abdata_joint"
- Now open the cpi data which includes inflation information. Perform all the necessary modifications to obtain a long format for only the UK.
- Merge the inflation data with the "abdata_joint".
- Reshape the data at the firm level. Reshape again at the year level.

Collapse

collapse takes the dataset in memory and creates a new dataset containing summary statistics of the original data. It is similar to pandas groupby.

Syntaxt



```
    collapse clist [if] [in] [weight] [, options]

where clist is either

            [(stat)] varlist [ [(stat)] ... ]
            [(stat)] target_var=varname [target_var=varname ...] [ [(stat)] ... ]
```

Exercice - Collapse

Using census5 from webuse:

- Load and describe the data set.
- Generate a new data set with the mean and the median of marriage and divorce by region.

Using the auto data:

- Generate a new data set with the average values of mpg and price and the maximum value of length for each rep78 and foreing value.
- Now reshape the data set from long to wide at the foreing level.

Exercice - Final

Use the acs small data:

- Analyze the relationship between race, sex, education level and earnings. You might want to use some of the graphs we learnt yesterday.
- Generate a data set with the mean of inctot and incwage for different race, sex, degree field. You can just keep indvididuals with education being college or higher.
- Reshape the data set from long to wide at the race level. Now on top of that do it at the sex level.

Just if there is more time:

- Save just the variables degree field and those related to incwage.
- Now reshape from long to wide and merge the data with the data set we just saved.