Introduction to Social Data Analytics

Week 5: Class 9

Arushi Kaushik

arkaushi@ucsd.edu

Today: Data Wrangling in Stata

By the end of today's lecture, you should be able to:

- Demonstrate appending and merging data
- Generate identifiers to differentiate between observations within a group
- Explain the difference between 1:1 and m:1 merges

Open class9.do if you haven't already.

Appending vs Merging

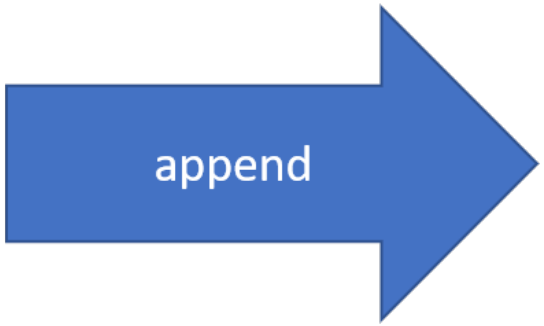
We append data to add observations, or rows.

We merge data to add variables, or columns.

Append to combine observations from tables with common variables

student	school	gpa
1	A	3.3
2	A	3.2

student	school	gpa
3	B	2.9
4	B	3.0

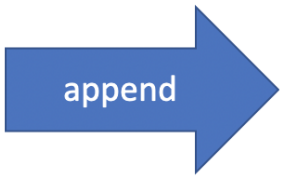


student	school	gpa
1	A	3.3
2	A	3.2
3	B	2.9
4	B	3.0

Append to combine observations from tables

student	school	gpa
1	A	3.3
2	A	3.2

student	school	gpa	scholarship
3	B	2.9	Yes
4	B	3	No



student	school	gpa	scholarship
1	A	3.3	.
2	A	3.2	.
3	B	2.9	Yes
4	B	3	No

If there are variables that exist **only in one** of the datasets, the datasets can still be appended

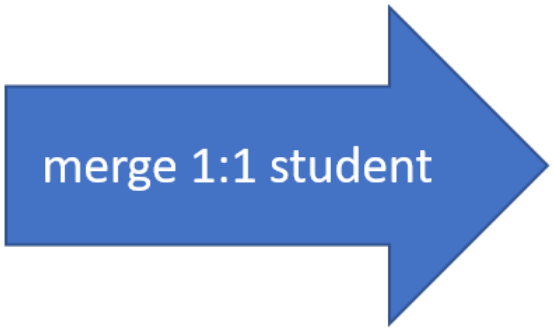
Class exercise: *append*

- Open `class9.do`
- Open `person2015` dataset
- `append using person2016`
- `append using person2017`
- save as `combined_worker.dta` file

Merge 1:1 when both tables have same unit of analysis

student	school
1	A
2	A
3	B
4	B

student	gpa
1	3.3
2	3.2
3	2.9
4	3.0



student	school	gpa
1	A	3.3
2	A	3.2
3	B	2.9
4	B	3.0

Class exercise: *merge 1:1*

- Open `class9.do`
- Open `demographics.dta` dataset

  We'll merge the `combined_worker.dta` with the demographics data

- Verify that **year-id** is the unique identifier in both datasets
- `sort year id`
- `merge 1:1 year id using "combined_worker.dta", gen(_merge)`
- save as `class9.dta` file

Merge m:1 when one table has a coarser unit of analysis

What is the unit of analysis of each table?

student	school	gpa
1	A	3.3
2	A	3.2
3	B	2.9
4	B	3.0

school	pop
A	1411
B	2692

student	school	gpa	pop
1	A	3.3	1411
2	A	3.2	1411
3	B	2.9	2692
4	B	3.0	2692

Merge 1:m when one table has a coarser unit of analysis

Depending upon which dataset is used as *master* and which one as *using*, we use `merge m:1` or `merge 1:m`

school	pop
A	1411
B	2692

student	school	gpa
1	A	3.3
2	A	3.2
3	B	2.9
4	B	3

school	student	pop	gpa
A	1	1411	3.3
A	2	1411	3.2
B	3	2692	2.9
B	4	2692	3

Merge 1:m when one table has a coarser unit of analysis

Depending upon which dataset is used as *master* and which one as *using*, we use `merge m:1` or `merge 1:m`

school	pop
A	1411
B	2692

student	school	gpa
1	A	3.3
2	A	3.2
3	B	2.9
4	B	3

school	student	pop	gpa
A	1	1411	3.3
A	2	1411	3.2
B	3	2692	2.9
B	4	2692	3

Use `merge 1:m` when *master* dataset has coarser unit of analysis than *using* data

Use `merge m:1` when *master* dataset has finer unit of analysis than *using* data.

Class exercise: *merge m:1 and 1:m*

- Open `class9.do`
- Open `class9.dta` dataset

  We'll merge the `class9.dta` with the `avg_annual_income.dta` in two ways.
  Verify that **year-id** is the unique identifier for the first dataset and **year** for the second one

- `merge m:1 school`
- `merge 1:m school`
- save as `class9_final.dta` file

Generating identifiers

1. Create a variable that reflects the unique identifier for `class9_final` dataset

1a. `gen unique_id1= _n`

1b. `tostring year, gen(yr_str)`

`tostring id gen(id_str)`

`gen unique_id2= yr_str + " " + id_str`

Generating identifiers (cont'd)

2. Creating a unique identifier **for** a group, e.g, year-race

2a. `egen id1= group(year white)`

2b. `gen id2= year*100 + white`

Generating identifiers (cont'd)

2. Creating a unique identifier **for** a group, e.g, year-race

2a. `egen id1= group(year white)`

2b. `gen id2= year*100 + white`

3. Creating a unique identifier **within** a group, e.g, year

3a. `bysort year:  gen within_id= _n`

Here are the commands/operators we covered today:

- `append`
- `merge 1:1; merge m:1`
- `_n`
- `bysort`
- `egen`
- `tostring`
- `group`