Data Transformation with Stata 15 Cheat Sheet

For more info see Stata's reference manual (stata.com)

Select Parts of Data (Subsetting)

Select Specific Columns

```
drop make
```
remove the 'make' variable

```
keep make price
```
opposite of drop; keep only variables 'make' and 'price'

Filter Specific Rows

```
drop if mpg < 20        drop in 1/4
```
drop observations based on a condition (left) or rows 1-4 (right)

```
keep in 1/30
```
opposite of drop; keep only rows 1-30

```
keep if inrange(price, 5000, 10000)
```
keep values of price between $5,000 – $10,000 (inclusive)

```
keep if inlist(make, "Honda Accord", "Honda Civic", "Subaru")
```
keep the specified values of make

```
sample 25
```
sample 25% of the observations in the dataset (use **set seed** # command for reproducible sampling)

Replace Parts of Data

Change Column Names

```
rename (rep78 foreign) (repairRecord carType)
```
rename one or multiple variables

Change Row Values

```
replace price = 5000 if price < 5000
```
replace all values of price that are less than $5,000 with 5000

```
recode price (0 / 5000 = 5000)
```
change all prices less than 5000 to be $5,000

```
recode foreign (0 = 2 "US")(1 = 1 "Not US"), gen(foreign2)
```
change the values and value labels then store in a new variable, foreign2

Replace Missing Values

```
mvdecode _all, mv(9999)
```
useful for cleaning survey datasets

replace the number 9999 with missing value in all variables

```
mvencode _all, mv(9999)
```
useful for exporting data

replace missing values with the number 9999 for all variables

Label Data

Value labels map string descriptions to numbers. They allow the underlying data to be numeric (making logical tests simpler) while also connecting the values to human-understandable text.

```
label define myLabel 0 "US" 1 "Not US"
label values foreign myLabel
```
define a label and apply it the values in foreign

```
label list
```
list all labels within the dataset

```
note: data note here
```
place note in dataset

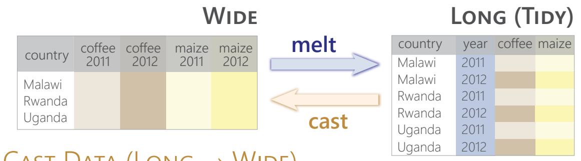
Reshape Data

```
webuse set https://github.com/GeoCenter/StataTraining/raw/master/Day2/Data
webuse "coffeeMaize.dta"
```
load demo dataset

Melt Data (Wide → Long)

```
reshape long coffee@ maize@, i(country) j(year)
```
reshape variables starting with coffee and maize · unique id variable (key) · create new variable which captures the info in the column names · new variable

convert a wide dataset to long



Tidy Datasets have each observation in its own row and each variable in its own

When datasets are tidy, they have a consistent, standard format that is easier to manipulate and analyze.

Cast Data (Long → Wide)

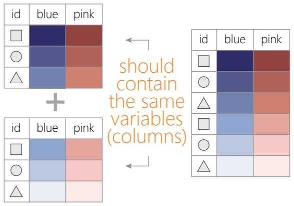
```
reshape wide coffee maize, i(country) j(year)
```
create new variables named coffee2011, maize2012... · what will be unique id variable (key) · create new variables with the year added to the column name

convert a long dataset to wide

```
xpose, clear varname
```
transpose rows and columns of data, clearing the data and saving old column names as a new variable called "_varname"

Combine Data

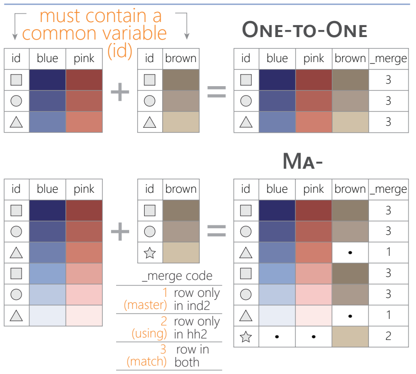
Adding (Appending) New Data



```
webuse coffeeMaize2.dta, clear
save coffeeMaize2.dta, replace
webuse coffeeMaize.dta, clear
```
load demo data

```
append using "coffeeMaize2.dta", gen(filenum)
```
add observations from "coffeeMaize2.dta" to current data and create variable "filenum" to track the origin of each observation

Merging Two Datasets Together



```
webuse ind_age.dta, clear
save ind_age.dta, replace
webuse ind_ag.dta, clear
```

```
merge 1:1 id using "ind_age.dta"
```
one-to-one merge of "ind_age.dta" into the loaded dataset and create variable "_merge" to track the origin

```
webuse hh2.dta, clear
save hh2.dta, replace
webuse ind2.dta, clear
```

```
merge m:1 hid using "hh2.dta"
```
many-to-one merge of "hh2.dta" into the loaded dataset and create variable "_merge" to track the origin

Fuzzy Matching: Combining Two Datasets without a Common ID

reclink match records from different data sets using probabilistic matching ssc install reclink

jarowinkler create distance measure for similarity between two strings ssc install jarowinkler

Manipulate Strings

Get String Properties

```
display length("This string has 29 characters")
```
return the length of the string

```
charlist make
```
* user-defined package

display the set of unique characters within a string

```
display strpos("Stata", "a")
```
return the position in Stata where a is first found

Find Matching Strings

```
display strmatch("123.89", "1??.?9")
```
return true (1) or false (0) if string matches pattern

```
display substr("Stata", 3, 5)
```
return string of 5 characters starting with position 3

```
list make if regexm(make, "[0-9]")
```
list observations where make matches the regular expression (here, records that contain a number)

```
list if regexm(make, "(Cad.|Chev.|Datsun)")
```
return all observations where make contains "Cad.", "Chev." or "Datsun"

compare the given list against the first word in make

```
list if inlist(word(make, 1), "Cad.", "Chev.", "Datsun")
```
return all observations where the first word of the make variable contains the listed words

Transform Strings

```
display regexr("My string", "My", "Your")
```
replace string1 ("My") with string2 ("Your")

```
replace make = subinstr(make, "Cad.", "Cadillac", 1)
```
replace first occurrence of "Cad." with Cadillac in the make variable

```
display stritrim(" Too much     Space")
```
replace consecutive spaces with a single space

```
display trim("   leading / trailing spaces    ")
```
remove extra spaces before and after a string

```
display strlower("STATA should not be ALL-CAPS")
```
change string case; see also **strupper**, **strproper**

```
display strtoname("1Var name")
```
convert string to Stata-compatible variable name

```
display real("100")
```
convert string to a numeric or missing value

Save & Export Data

```
compress
```
compress data in memory

```
save "myData.dta", replace
saveold "myData.dta", replace version(12)
```
Stata 12-compatible file

save data in Stata format, replacing the data if a file with same name exists

```
export excel "myData.xls", /*
 */ firstrow(variables) replace
```
export data as an Excel file (.xls) with the variable names as the first row

```
export delimited "myData.csv", delimiter(",") replace
```
export data as a comma-delimited file (.csv)

Tim Essam (tessam@usaid.gov) • Laura Hughes (lhughes@usaid.gov) · follow us @StataRGIS and @flaneuseks

inspired by RStudio's awesome Cheat Sheets (rstudio.com/resources/cheatsheets)

geocenter.github.io/StataTraining

Disclaimer: we are not affiliated with Stata. But we like it.

updated June 2016