Grammar of Data Transformation

K Arnold, based on IntroDS.org

Week 3

- Lab 2 end-of-day today
- Discussion 1 replies by tomorrow
- Hw 2 due Wednesday
    - Check that your `.md` file is on GitHub
    - What does each row represent?
    - How many rows *should* there be?
- Office Hours: [kca] Mon 8-9am, Fri 3-4pm; [yk] Wed 4:30-5:30pm

Question-answers

> Am I submitting my labs and homework correctly?

- See checklist at end of Lab 1.
- Main point: Check your `.md` file on GitHub.

> When will we get feedback?

- General feedback already given in class.
- A bit backed up on specific feedback.

> Can we make animated plots like in the video/

- Stay tuned for Plotly.

Questions for you

- How is week 3 of Fall 2020?
- What's working well in DATA 202? What's challenging?

Questions for you

- How is week 3 of Fall 2020?
- What's working well in DATA 202? What's challenging?

- How are the readings and prep exercises?
- How long are you spending on labs outside of class?
- How hard are labs compared with homework?

So far

- *R/RStudio/Rmarkdown/Git*: a toolkit for reproducible collaborative analysis and reporting
- `ggplot2`: a *Grammar of Graphics*
  - a language for describing, and building, visualizations
  - concepts apply to many other toolkits

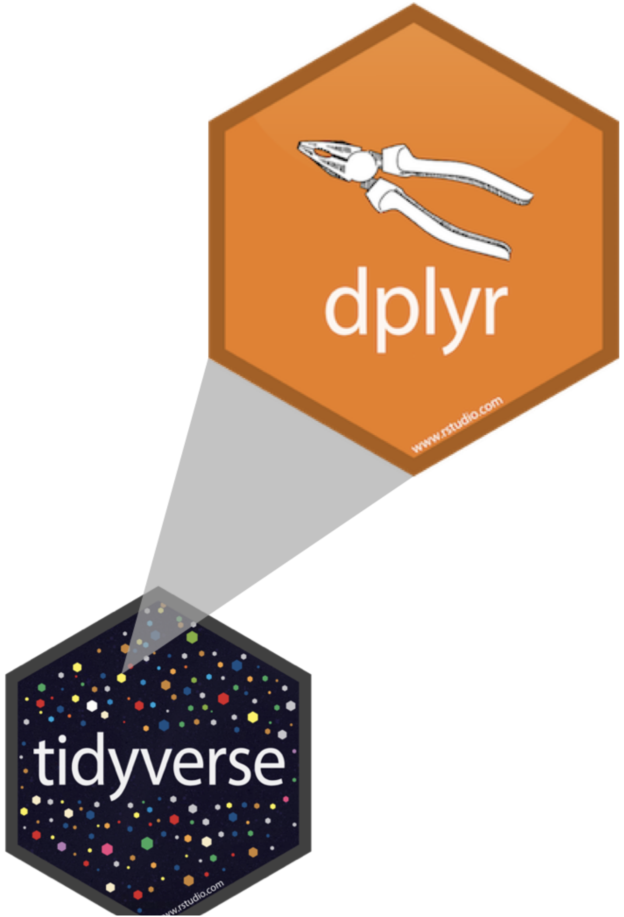
This week:

- `dplyr`: a *Grammar of Data Transformation*
  - basic concepts will show up again in Python (Pandas) and SQL.

Data wrangling and summarizing with *dplyr*

A grammar of data wrangling

Functions as verbs that manipulate data frames



- `select`: pick columns by name
- `arrange`: reorder rows
- `slice`: pick rows by index(es)
- `slice_sample`: randomly sample rows
- `filter`: pick rows matching criteria
- `distinct`: filter for unique rows
- `mutate`: add new variables
- `summarize`: reduce variables to values

Rules of *dplyr* functions

- First argument is *always* a data frame
- Subsequent arguments say what to do with that data frame
- Always return a data frame
- Don't modify in place

Bike crashes in NC 2007 - 2014

```
ncbikecrash <- read_csv("data/ncbikecrash.csv")
```

```
glimpse(ncbikecrash)
```

```
## Rows: 7,467
## Columns: 53
## $ object_id            <dbl> 1686, 1674, 1673, 1687, 1653, 1665, 1642, 1675, …
## $ city                 <chr> "None - Rural Crash", "Henderson", "None - Rural…
## $ county               <chr> "Wayne", "Vance", "Lincoln", "Columbus", "New Ha…
## $ region               <chr> "Coastal", "Piedmont", "Piedmont", "Coastal", "C…
## $ development          <chr> "Farms, Woods, Pastures", "Residential", "Farms,…
## $ locality             <chr> "Rural (<30% Developed)", "Mixed (30% To 70% Dev…
## $ on_road              <chr> "SR 1915", "NICHOLAS ST", "US 321", "W BURKHEAD …
## $ rural_urban          <chr> "Rural", "Urban", "Rural", "Urban", "Urban", "Ru…
## $ speed_limit          <chr> "50 - 55  MPH", "30 - 35  MPH", "50 - 55  MPH", …
## $ traffic_control      <chr> "No Control Present", "Stop Sign", "Double Yello…
## $ weather              <chr> "Clear", "Clear", "Clear", "Rain", "Clear", "Clo…
## $ workzone             <chr> "No", "No", "No", "No", "No", "No", "No", "No", …
## $ bike_age             <chr> "52", "66", "33", "52", "22", "15", "41", "14", …
## $ bike_age_group       <chr> "50-59", "60-69", "30-39", "50-59", "20-24", "11…
## $ bike_alcohol         <chr> "No", "No", "No", "Yes", "No", "No", "No", "No",…
```

Variables

View the names of variables via

```
names(ncbikecrash)
```

```
##  [1] "object_id"            "city"                "county"
##  [4] "region"               "development"         "locality"
##  [7] "on_road"              "rural_urban"         "speed_limit"
## [10] "traffic_control"      "weather"             "workzone"
## [13] "bike_age"             "bike_age_group"      "bike_alcohol"
## [16] "bike_alcohol_drugs"   "bike_direction"      "bike_injury"
## [19] "bike_position"        "bike_race"           "bike_sex"
## [22] "driver_age"           "driver_age_group"    "driver_alcohol"
## [25] "driver_alcohol_drugs" "driver_est_speed"    "driver_injury"
## [28] "driver_race"          "driver_sex"          "driver_vehicle_type"
## [31] "crash_alcohol"        "crash_date"          "crash_day"
## [34] "crash_group"          "crash_hour"          "crash_location"
## [37] "crash_month"          "crash_severity"      "crash_time"
## [40] "crash_type"           "crash_year"          "ambulance_req"
## [43] "hit_run"              "light_condition"     "road_character"
## [46] "road_class"           "road_condition"      "road_configuration"
## [49] "road_defects"         "road_feature"        "road_surface"
```

Select columns

```
select(ncbikecrash, county, bike_age)
```

```
## # A tibble: 7,467 x 2
##    county      bike_age
##    <chr>       <chr>
##  1 Wayne       52
##  2 Vance       66
##  3 Lincoln     33
##  4 Columbus    52
##  5 New Hanover 22
##  6 Robeson     15
##  7 Richmond    41
##  8 Wake        14
##  9 Columbus    16
## 10 Craven      54
## # … with 7,457 more rows
```

Select columns

```
select(ncbikecrash, county, bike_age)
```

```
## # A tibble: 7,467 x 2
##    county      bike_age
##    <chr>       <chr>
##  1 Wayne       52
##  2 Vance       66
##  3 Lincoln     33
##  4 Columbus    52
##  5 New Hanover 22
##  6 Robeson     15
##  7 Richmond    41
##  8 Wake        14
##  9 Columbus    16
## 10 Craven      54
## # … with 7,457 more rows
```

What if we wanted to select these columns, and then arrange the data in ascending order of biker age?

Data wrangling, step-by-step

Select:

```
ncbikecrash %>%
  select(county, bike_age)
```

```
## # A tibble: 7,467 x 2
##    county      bike_age
##    <chr>       <chr>
##  1 Wayne       52
##  2 Vance       66
##  3 Lincoln     33
##  4 Columbus    52
##  5 New Hanover 22
##  6 Robeson     15
##  7 Richmond    41
##  8 Wake        14
##  9 Columbus    16
## 10 Craven      54
## # … with 7,457 more rows
```

Select, then arrange:

```
ncbikecrash %>%
  select(county, bike_age) %>%
  arrange(bike_age)
```

```
## # A tibble: 7,467 x 2
##    county      bike_age
##    <chr>       <chr>
##  1 New Hanover 0
##  2 Carteret    1
##  3 Guilford    1
##  4 Pitt        10
##  5 Cumberland  10
##  6 Carteret    10
##  7 Hoke        10
##  8 Martin      10
##  9 New Hanover 10
## 10 Onslow      10
## # … with 7,457 more rows
```

Pipes

What is a pipe?

In programming, a pipe is a technique for passing information from one process to another.

What is a pipe?

In programming, a pipe is a technique for passing information from one process to another.

- Start with the data frame `ncbikecrash`

```
ncbikecrash %>%
  select(county, bike_age) %>%
  arrange(bike_age)
```

```
## # A tibble: 7,467 x 2
##    county      bike_age
##    <chr>       <chr>
##  1 New Hanover 0
##  2 Carteret    1
##  3 Guilford    1
##  4 Pitt        10
##  5 Cumberland  10
##  6 Carteret    10
##  7 Hoke        10
##  8 Martin      10
##  9 New Hanover 10
## 10 Onslow      10
## # … with 7,457
```

What is a pipe?

In programming, a pipe is a technique for passing information from one process to another.

- Start with the data frame ncbikecrash,
- then we select the variables county and bike_age,

```
ncbikecrash %>%
  select(county, bike_age) %>%
  arrange(bike_age)
```

```
## # A tibble: 7,467 x 2
##    county      bike_age
##    <chr>       <chr>
##  1 New Hanover 0
##  2 Carteret    1
##  3 Guilford    1
##  4 Pitt        10
##  5 Cumberland  10
##  6 Carteret    10
##  7 Hoke        10
##  8 Martin      10
##  9 New Hanover 10
## 10 Onslow      10
## # … with 7,457
```

What is a pipe?

In programming, a pipe is a technique for passing information from one process to another.

- Start with the data frame `ncbikecrash`,
- then we `select` the variables `county` and `bike_age`,
- and then we `arrange` the data frame by `bike_age` in ascending order.

```
ncbikecrash %>%
  select(county, bike_age) %>%
  arrange(bike_age)
```

```
## # A tibble: 7,467 x 2
##    county      bike_age
##    <chr>       <chr>
##  1 New Hanover 0
##  2 Carteret    1
##  3 Guilford    1
##  4 Pitt        10
##  5 Cumberland  10
##  6 Carteret    10
##  7 Hoke        10
##  8 Martin      10
##  9 New Hanover 10
## 10 Onslow      10
```

How does a pipe work?

Conventional (nested functions):

```
arrange(select(ncbikecrash, county, bike_age), bike_age)
```

With pipes:

```
ncbikecrash %>%
  select(county, bike_age) %>%
  arrange(bike_age)
```

What about other arguments?

Use the dot (`.`) to

- send results to a function argument other than first one or
- use the previous result for multiple arguments

```
starwars %>%
  filter(., species == "Human") %>%
  lm(mass ~ height, data = .)
```

```
##
## Call:
## lm(formula = mass ~ height, data = .)
##
## Coefficients:
## (Intercept)       height
##     -116.58         1.11
```

A note on piping and layering

- The `%>%` operator in **dplyr** functions is called the *pipe* operator. This means you "pipe" the output of the previous line of code as the first input of the next line of code.
- The `+` operator in **ggplot2** functions is used for "*layering*". This means you create the plot in layers, separated by `+`.
- Many of the styling principles are consistent across `%>%` and `+`:
  - always a space before
  - always a line break after (for pipelines with more than 2 lines)

Data wrangling with dplyr

Exercise: Hotel Wrangling

select to keep variables

```
ncbikecrash %>%
  select(locality, speed_limit)
```

```
## # A tibble: 7,467 x 2
##    locality                     speed_limit
##    <chr>                        <chr>
##  1 Rural (<30% Developed)       50 - 55  MPH
##  2 Mixed (30% To 70% Developed) 30 - 35  MPH
##  3 Rural (<30% Developed)       50 - 55  MPH
##  4 Urban (>70% Developed)       30 - 35  MPH
##  5 Urban (>70% Developed)       <NA>
##  6 Rural (<30% Developed)       50 - 55  MPH
##  7 Mixed (30% To 70% Developed) 30 - 35  MPH
##  8 Urban (>70% Developed)       30 - 35  MPH
##  9 Rural (<30% Developed)       30 - 35  MPH
## 10 Urban (>70% Developed)       20 - 25  MPH
## # … with 7,457 more rows
```

select to exclude variables

```r
ncbikecrash %>%
  select(-object_id)
```

```
## # A tibble: 7,467 x 52
##    city  county region development locality on_road rural_urban speed_limit
##    <chr> <chr>  <chr>  <chr>       <chr>    <chr>   <chr>       <chr>      
##  1 None… Wayne  Coast… Farms, Woo… Rural (… SR 1915 Rural       50 - 55  M…
##  2 Hend… Vance  Piedm… Residential Mixed (… NICHOL… Urban       30 - 35  M…
##  3 None… Linco… Piedm… Farms, Woo… Rural (… US 321  Rural       50 - 55  M…
##  4 Whit… Colum… Coast… Commercial  Urban (… W BURK… Urban       30 - 35  M…
##  5 Wilm… New H… Coast… Residential Urban (… RACINE… Urban       <NA>       
##  6 None… Robes… Coast… Farms, Woo… Rural (… SR 1513 Rural       50 - 55  M…
##  7 None… Richm… Piedm… Residential Mixed (… SR 1903 Rural       30 - 35  M…
##  8 Rale… Wake   Piedm… Commercial  Urban (… PERSON… Urban       30 - 35  M…
##  9 Whit… Colum… Coast… Residential Rural (… FLOWER… Urban       30 - 35  M…
## 10 New … Craven Coast… Residential Urban (… SUTTON… Urban       20 - 25  M…
## # … with 7,457 more rows, and 44 more variables: traffic_control <chr>,
## #   weather <chr>, workzone <chr>, bike_age <chr>, bike_age_group <chr>,
## #   bike_alcohol <chr>, bike_alcohol_drugs <chr>, bike_direction <chr>,
## #   bike_injury <chr>, bike_position <chr>, bike_race <chr>, bike_sex <chr>,
## #   driver_age <chr>, driver_age_group <chr>, driver_alcohol <chr>,
## #   driver_alcohol_drugs <chr>, driver_est_speed <chr>, driver_injury <chr>,
## #   driver_race <chr>, driver_sex <chr>, driver_vehicle_type <chr>
```

`select` variables with certain characteristics

```r
ncbikecrash %>%
  select(starts_with("bike_"))
```

```
## # A tibble: 7,467 x 9
##    bike_age bike_age_group bike_alcohol bike_alcohol_dr… bike_direction
##    <chr>    <chr>          <chr>        <chr>            <chr>
##  1 52       50-59          No           <NA>             With Traffic
##  2 66       60-69          No           <NA>             With Traffic
##  3 33       30-39          No           <NA>             With Traffic
##  4 52       50-59          Yes          <NA>             <NA>
##  5 22       20-24          No           <NA>             Facing Traffic
##  6 15       11-15          No           <NA>             With Traffic
##  7 41       40-49          No           <NA>             Facing Traffic
##  8 14       11-15          No           <NA>             <NA>
##  9 16       16-19          No           <NA>             Facing Traffic
## 10 54       50-59          No           <NA>             With Traffic
## # … with 7,457 more rows, and 4 more variables: bike_injury <chr>,
## #   bike_position <chr>, bike_race <chr>, bike_sex <chr>
```

select variables with certain characteristics

```
ncbikecrash %>%
  select(ends_with("age"))
```

```
## # A tibble: 7,467 x 2
##    bike_age driver_age
##    <chr>    <chr>
##  1 52       34
##  2 66       <NA>
##  3 33       37
##  4 52       55
##  5 22       25
##  6 15       17
##  7 41       <NA>
##  8 14       50
##  9 16       32
## 10 54       69
## # … with 7,457 more rows
```

Select helpers

- `starts_with()`: Starts with a prefix
- `ends_with()`: Ends with a suffix
- `contains()`: Contains a literal string
- `num_range()`: Matches a numerical range like x01, x02, x03
- `one_of()`: Matches variable names in a character vector
- `everything()`: Matches all variables
- `last_col()`: Select last variable, possibly with an offset
- `matches()`: Matches a regular expression (a sequence of symbols/characters expressing a string/pattern to be searched for within text)

See help for any of these functions for more info, e.g. `?everything`.

`arrange` in ascending / descending order

```r
ncbikecrash %>%
  select(ends_with("age")) %>%
  arrange(bike_age)
```

```
## # A tibble: 7,467 x 2
##    bike_age driver_age
##    <chr>    <chr>
##  1 0        47
##  2 1        70+
##  3 1        61
##  4 10       30
##  5 10       19
##  6 10       22
##  7 10       18
##  8 10       27
##  9 10       53
## 10 10       <NA>
## # … with 7,457 more rows
```

```r
ncbikecrash %>%
  select(ends_with("age")) %>%
  arrange(desc(bike_age))
```

```
## # A tibble: 7,467 x 2
##    bike_age driver_age
##    <chr>    <chr>
##  1 9        23
##  2 9        35
##  3 9        70+
##  4 9        41
##  5 9        53
##  6 9        18
##  7 9        45
##  8 9        19
##  9 9        70+
## 10 9        59
## # … with 7,457 more rows
```

`slice` for certain row numbers

First five

```r
ncbikecrash %>%
  slice(1:5)
```

```
## # A tibble: 5 x 53
##   object_id city  county region development locality on_road rural_urban
##       <dbl> <chr> <chr>  <chr>  <chr>       <chr>    <chr>   <chr>      
## 1      1686 None… Wayne  Coast… Farms, Woo… Rural (… SR 1915 Rural      
## 2      1674 Hend… Vance  Piedm… Residential Mixed (… NICHOL… Urban      
## 3      1673 None… Linco… Piedm… Farms, Woo… Rural (… US 321  Rural      
## 4      1687 Whit… Colum… Coast… Commercial  Urban (… W BURK… Urban      
## 5      1653 Wilm… New H… Coast… Residential Urban (… RACINE… Urban      
## # … with 45 more variables: speed_limit <chr>, traffic_control <chr>,
## #   weather <chr>, workzone <chr>, bike_age <chr>, bike_age_group <chr>,
## #   bike_alcohol <chr>, bike_alcohol_drugs <chr>, bike_direction <chr>,
## #   bike_injury <chr>, bike_position <chr>, bike_race <chr>, bike_sex <chr>,
## #   driver_age <chr>, driver_age_group <chr>, driver_alcohol <chr>,
## #   driver_alcohol_drugs <chr>, driver_est_speed <chr>, driver_injury <chr>,
## #   driver_race <chr>, driver_sex <chr>, driver_vehicle_type <chr>,
## #   crash_alcohol <chr>, crash_date <chr>, crash_day <chr>, crash_group <chr>,
```

`slice` for certain row numbers

Last five

```r
last_row <- nrow(ncbikecrash)
ncbikecrash %>%
  slice((last_row - 4):last_row)
```

```
## # A tibble: 5 x 53
##   object_id city  county region development locality on_road rural_urban
##       <dbl> <chr> <chr>  <chr>  <chr>       <chr>    <chr>   <chr>
## 1      6989 High… Guilf… Piedm… Residential Urban (… <NA>    Urban
## 2      6991 Wilm… New H… Coast… Residential Urban (… <NA>    Urban
## 3      6995 Kins… Lenoir Coast… Commercial  Urban (… <NA>    Urban
## 4      6998 Faye… Cumbe… Coast… Residential Urban (… <NA>    Urban
## 5      7000 None… Onslow Coast… Farms, Woo… Rural (… <NA>    Rural
## # … with 45 more variables: speed_limit <chr>, traffic_control <chr>,
## #   weather <chr>, workzone <chr>, bike_age <chr>, bike_age_group <chr>,
## #   bike_alcohol <chr>, bike_alcohol_drugs <chr>, bike_direction <chr>,
## #   bike_injury <chr>, bike_position <chr>, bike_race <chr>, bike_sex <chr>,
## #   driver_age <chr>, driver_age_group <chr>, driver_alcohol <chr>,
## #   driver_alcohol_drugs <chr>, driver_est_speed <chr>, driver_injury <chr>,
## #   driver_race <chr>, driver_sex <chr>, driver_vehicle_type <chr>,
```

`sample_n` / `sample_frac` for a random sample

- `slice_sample`: randomly sample `n = 5` observations

```r
ncbikecrash_n5 <- ncbikecrash %>%
  slice_sample(n = 5, replace = FALSE)
dim(ncbikecrash_n5)
```

```
## [1]  5 53
```

`sample_n` / `sample_frac` for a random sample

- `slice_sample`: randomly sample `n = 5` observations

```
ncbikecrash_n5 <- ncbikecrash %>%
  slice_sample(n = 5, replace = FALSE)
dim(ncbikecrash_n5)
```

```
## [1]  5 53
```

- `sample_frac`: randomly sample `prop = 20%` of observations

```
ncbikecrash_perc20 <-ncbikecrash %>%
  slice_sample(prop = 0.2, replace = FALSE)
dim(ncbikecrash_perc20)
```

```
## [1] 1493   53
```

`filter` to select a subset of rows

Crashes in Durham County

```r
ncbikecrash %>%
  filter(county == "Durham")
```

```
## # A tibble: 340 x 53
##    object_id city  county region development locality on_road rural_urban
##        <dbl> <chr> <chr>  <chr>  <chr>       <chr>    <chr>   <chr>
##  1      2452 Durh… Durham Piedm… Residential Urban (… <NA>    Urban
##  2      2441 Durh… Durham Piedm… Commercial  Urban (… <NA>    Urban
##  3      2466 Durh… Durham Piedm… Commercial  Urban (… <NA>    Urban
##  4       549 Durh… Durham Piedm… Residential Urban (… PARK A… Urban
##  5       598 Durh… Durham Piedm… Residential Urban (… BELT S… Urban
##  6       603 Durh… Durham Piedm… Residential Urban (… HINSON… Urban
##  7      3974 Durh… Durham Piedm… Commercial  Urban (… <NA>    Urban
##  8      7134 Durh… Durham Piedm… Commercial  Urban (… <NA>    Urban
##  9      1670 Durh… Durham Piedm… Commercial  Urban (… INFINI… Urban
## 10      1773 Durh… Durham Piedm… Residential Urban (… <NA>    Urban
## # … with 330 more rows, and 45 more variables: speed_limit <chr>,
## #   traffic_control <chr>, weather <chr>, workzone <chr>, bike_age <chr>,
## #   bike_age_group <chr>, bike_alcohol <chr>, bike_alcohol_drugs <chr>,
```

`filter` for many conditions at once

Crashes in Durham County where biker is 0-5 years old

```r
ncbikecrash %>%
  filter(
    county == "Durham",
    bike_age_group == "0-5"
    )
```

```
## # A tibble: 4 x 53
##   object_id city  county region development locality on_road rural_urban
##       <dbl> <chr> <chr>  <chr>  <chr>       <chr>    <chr>   <chr>      
## 1      4062 Durh… Durham Piedm… Residential Urban (… <NA>    Urban      
## 2       414 Durh… Durham Piedm… Residential Urban (… PVA 90… Urban      
## 3      3016 Durh… Durham Piedm… Residential Urban (… <NA>    Urban      
## 4      1383 Durh… Durham Piedm… Residential Urban (… PVA 62… Urban      
## # … with 45 more variables: speed_limit <chr>, traffic_control <chr>,
## #   weather <chr>, workzone <chr>, bike_age <chr>, bike_age_group <chr>,
## #   bike_alcohol <chr>, bike_alcohol_drugs <chr>, bike_direction <chr>,
## #   bike_injury <chr>, bike_position <chr>, bike_race <chr>, bike_sex <chr>,
## #   driver_age <chr>, driver_age_group <chr>, driver_alcohol <chr>,
## #   driver_alcohol_drugs <chr>, driver_est_speed <chr>, driver_injury <chr>,
```

Logical operators in R

operator	definition	operator	definition
<	less than	x | y	x OR y
<=	less than or equal to	is.na(x)	test if x is NA
>	greater than	!is.na(x)	test if x is not NA
>=	greater than or equal to	x %in% y	test if x is in y
==	exactly equal to	!(x %in% y)	test if x is not in y
!=	not equal to	!x	not x
x & y	x AND y		

Fill in the blanks for filtering for crashes **not** in Durham County where crash year is after 2014 and `bike_position` is not `NA`.

```
ncbikecrash %>%
  filter(
    county ____ "Durham",
    crash_year ____ 2014,
    ____
    )
```

Fill in the blanks for filtering for crashes **not** in Durham County where crash year is after 2014 and `bike_position` is not `NA`.

```
ncbikecrash %>%
  filter(
    county != "Durham",
    crash_year > 2014,
    !is.na(bike_position)
    )
```

```
## # A tibble: 0 x 53
## # … with 53 variables: object_id <dbl>, city <chr>, county <chr>, region <chr>,
## #   development <chr>, locality <chr>, on_road <chr>, rural_urban <chr>,
## #   speed_limit <chr>, traffic_control <chr>, weather <chr>, workzone <chr>,
## #   bike_age <chr>, bike_age_group <chr>, bike_alcohol <chr>,
## #   bike_alcohol_drugs <chr>, bike_direction <chr>, bike_injury <chr>,
## #   bike_position <chr>, bike_race <chr>, bike_sex <chr>, driver_age <chr>,
## #   driver_age_group <chr>, driver_alcohol <chr>, driver_alcohol_drugs <chr>,
## #   driver_est_speed <chr>, driver_injury <chr>, driver_race <chr>,
## #   driver_sex <chr>, driver_vehicle_type <chr>, crash_alcohol <chr>,
## #   crash_date <chr>, crash_day <chr>, crash_group <chr>, crash_hour <dbl>,
## #   crash_location <chr>, crash_month <chr>, crash_severity <chr>,
## #   crash_time <time>, crash_type <chr>, crash_year <dbl>, ambulance_req <chr>,
## #   hit_run <chr>, light_condition <chr>, road_character <chr>,
## #   road_class <chr>, road_condition <chr>, road_configuration <chr>,
## #   road_defects <chr>, road_feature <chr>, road_surface <chr>,
## #   num_lanes <chr>, geo_point <chr>
```

distinct to filter for unique rows

… and arrange to order alphabetically

```
ncbikecrash %>%
  distinct(county) %>%
  arrange(county)
```

```
## # A tibble: 101 x 1
##    county
##    <chr>
##  1 Alamance
##  2 Alexander
##  3 Alleghany
##  4 Anson
##  5 Ashe
##  6 Avery
##  7 Beaufort
##  8 Bertie
##  9 Bladen
## 10 Brunswick
## # … with 91 more rows
```

```
ncbikecrash %>%
  select(county, city) %>%
  distinct() %>%
  arrange(county, city)
```

```
## # A tibble: 391 x 2
##    county    city
##    <chr>     <chr>
##  1 Alamance  Alamance
##  2 Alamance  Burlington
##  3 Alamance  Elon
##  4 Alamance  Elon College
##  5 Alamance  Gibsonville
##  6 Alamance  Graham
##  7 Alamance  Green Level
##  8 Alamance  Mebane
##  9 Alamance  None - Rural Crash
## 10 Alexander None - Rural Crash
## # … with 381 more rows
```

Code Style

> "Good coding style is like correct punctuation: you can manage without it, butitsuremakesthingseasiertoread."
> Hadley Wickham

- Recommended: Tidyverse style guide https://style.tidyverse.org/

Summary

- File names and code chunks: `data-wrangling`, not `Data Wrangling`.
- Variable names: `hourly_rides`, not `hourlyRides` or `hourly.rides` or `rides_by_hour_with_weather`
    - Informative but short. Don't reuse.

Spacing

- Put a space before and after all infix operators (=, +, -, <-, etc.), and when naming arguments in function calls
- Always put a space after a comma, and never before (just like in regular English)

```r
# Good
average <- mean(feet / 12 + inches, na.rm = TRUE)

# Bad
average<-mean(feet/12+inches,na.rm=TRUE)
```

ggplot2

- Always end a line with `+`
- Always indent the next line

```
# Good
ggplot(diamonds, mapping = aes(x = price)) +
  geom_histogram()

# Bad
ggplot(diamonds,mapping=aes(x=price))+geom_histogram()
```

Tidy data

Tidy data

> Happy families are all alike; every unhappy family is unhappy in its own way.
> Leo Tolstoy

Tidy data

> Happy families are all alike; every unhappy family is unhappy in its own way.
> Leo Tolstoy

Characteristics of tidy data:

- Each variable forms a column.
- Each observation forms a row.
- Each type of observational unit forms a table.

Tidy data

> Happy families are all alike; every unhappy family is unhappy in its own way.
> Leo Tolstoy

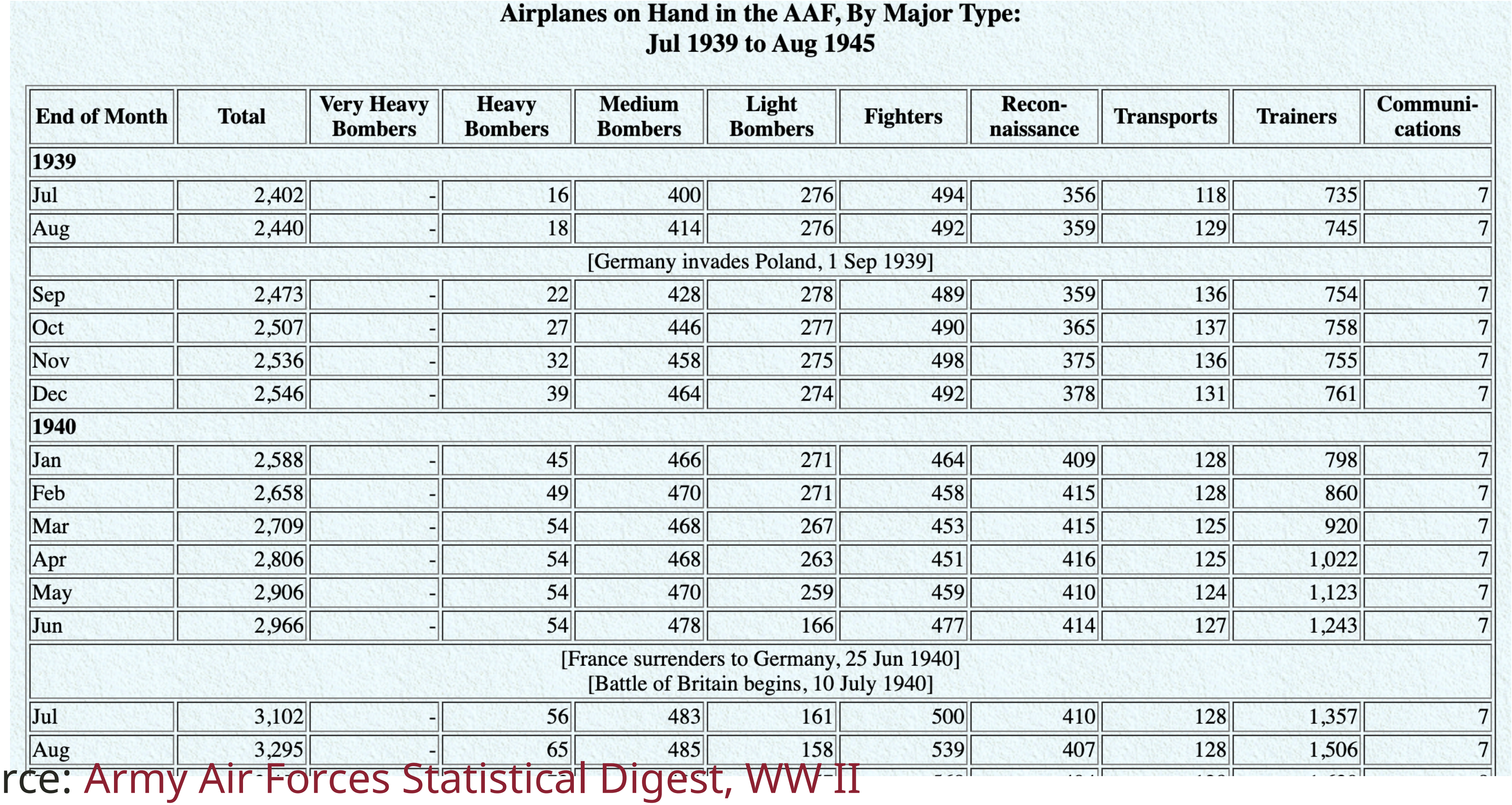
Characteristics of tidy data:

- Each variable forms a column.
- Each observation forms a row.
- Each type of observational unit forms a table.

Characteristics of untidy data:

- Varies.

What makes this data not tidy?

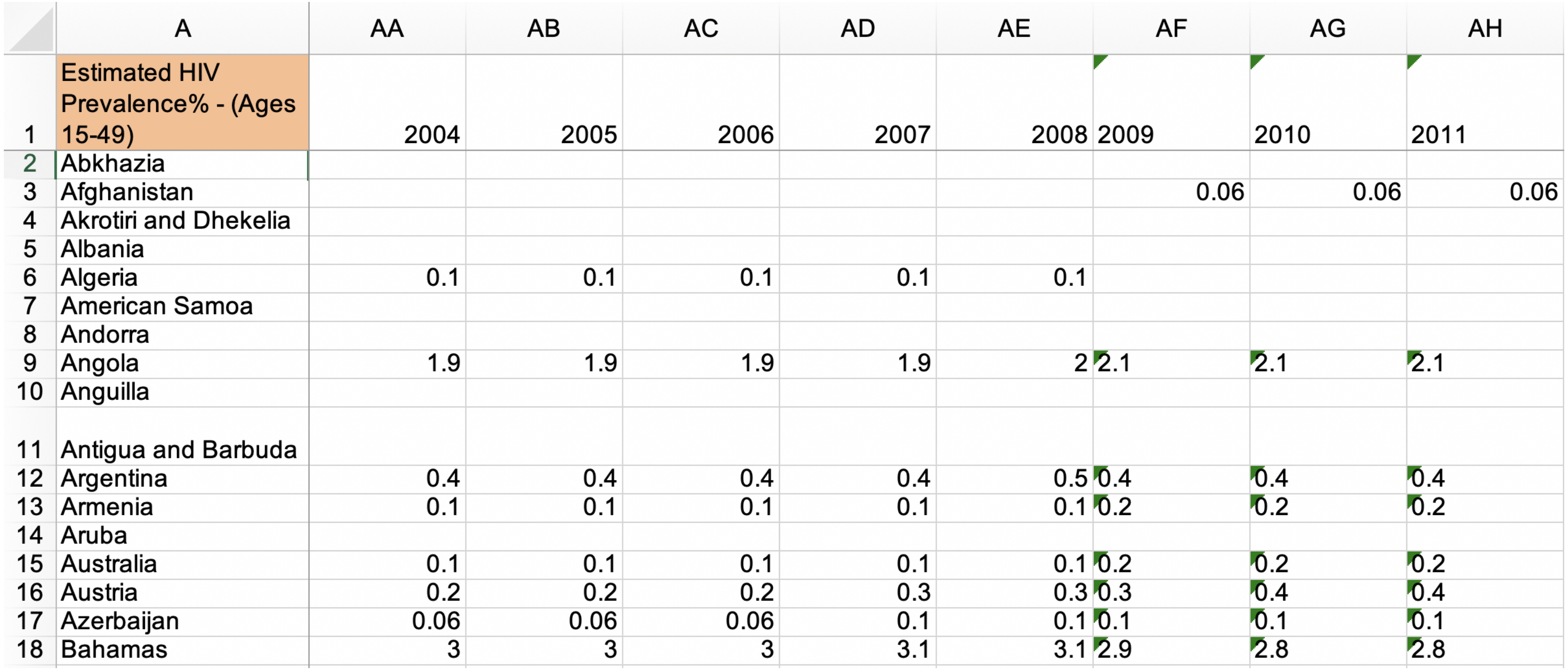


Airplanes on Hand in the AAF, By Major Type:
Jul 1939 to Aug 1945

End of Month	Total	Very Heavy Bombers	Heavy Bombers	Medium Bombers	Light Bombers	Fighters	Recon-naissance	Transports	Trainers	Communi-cations
1939										
Jul	2,402	-	16	400	276	494	356	118	735	7
Aug	2,440	-	18	414	276	492	359	129	745	7
[Germany invades Poland, 1 Sep 1939]										
Sep	2,473	-	22	428	278	489	359	136	754	7
Oct	2,507	-	27	446	277	490	365	137	758	7
Nov	2,536	-	32	458	275	498	375	136	755	7
Dec	2,546	-	39	464	274	492	378	131	761	7
1940										
Jan	2,588	-	45	466	271	464	409	128	798	7
Feb	2,658	-	49	470	271	458	415	128	860	7
Mar	2,709	-	54	468	267	453	415	125	920	7
Apr	2,806	-	54	468	263	451	416	125	1,022	7
May	2,906	-	54	470	259	459	410	124	1,123	7
Jun	2,966	-	54	478	166	477	414	127	1,243	7
[France surrenders to Germany, 25 Jun 1940] [Battle of Britain begins, 10 July 1940]										
Jul	3,102	-	56	483	161	500	410	128	1,357	7
Aug	3,295	-	65	485	158	539	407	128	1,506	7

Source: Army Air Forces Statistical Digest, WW II

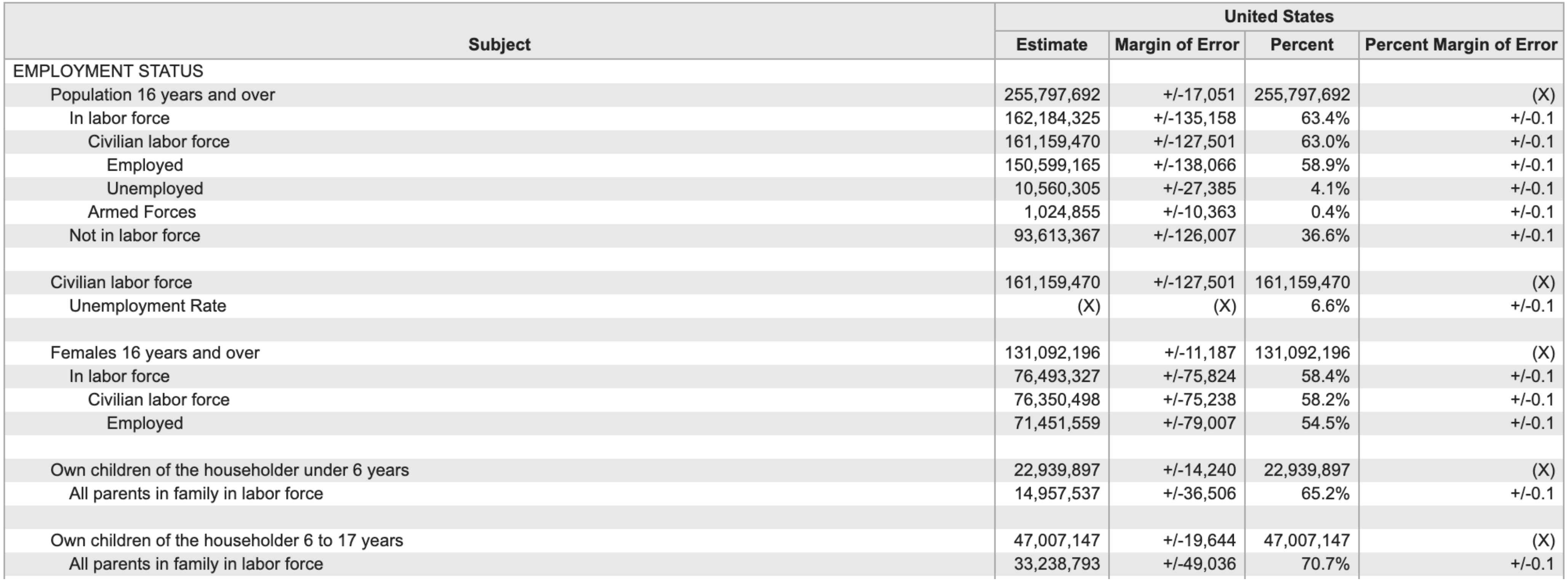
What makes this data not tidy?



	A	AA	AB	AC	AD	AE	AF	AG	AH
1	Estimated HIV Prevalence% - (Ages 15-49)	2004	2005	2006	2007	2008	2009	2010	2011
2	Abkhazia								
3	Afghanistan						0.06	0.06	0.06
4	Akrotiri and Dhekelia								
5	Albania								
6	Algeria	0.1	0.1	0.1	0.1	0.1			
7	American Samoa								
8	Andorra								
9	Angola	1.9	1.9	1.9	1.9	2	2.1	2.1	2.1
10	Anguilla								
11	Antigua and Barbuda								
12	Argentina	0.4	0.4	0.4	0.4	0.5	0.4	0.4	0.4
13	Armenia	0.1	0.1	0.1	0.1	0.1	0.2	0.2	0.2
14	Aruba								
15	Australia	0.1	0.1	0.1	0.1	0.1	0.2	0.2	0.2
16	Austria	0.2	0.2	0.2	0.3	0.3	0.3	0.4	0.4
17	Azerbaijan	0.06	0.06	0.06	0.1	0.1	0.1	0.1	0.1
18	Bahamas	3	3	3	3.1	3.1	2.9	2.8	2.8

Source: Gapminder, Estimated HIV prevalence among 15-49 year olds

What makes this data not tidy?



Subject	United States			
	Estimate	Margin of Error	Percent	Percent Margin of Error
EMPLOYMENT STATUS				
Population 16 years and over	255,797,692	+/-17,051	255,797,692	(X)
In labor force	162,184,325	+/-135,158	63.4%	+/-0.1
Civilian labor force	161,159,470	+/-127,501	63.0%	+/-0.1
Employed	150,599,165	+/-138,066	58.9%	+/-0.1
Unemployed	10,560,305	+/-27,385	4.1%	+/-0.1
Armed Forces	1,024,855	+/-10,363	0.4%	+/-0.1
Not in labor force	93,613,367	+/-126,007	36.6%	+/-0.1
Civilian labor force	161,159,470	+/-127,501	161,159,470	(X)
Unemployment Rate	(X)	(X)	6.6%	+/-0.1
Females 16 years and over	131,092,196	+/-11,187	131,092,196	(X)
In labor force	76,493,327	+/-75,824	58.4%	+/-0.1
Civilian labor force	76,350,498	+/-75,238	58.2%	+/-0.1
Employed	71,451,559	+/-79,007	54.5%	+/-0.1
Own children of the householder under 6 years	22,939,897	+/-14,240	22,939,897	(X)
All parents in family in labor force	14,957,537	+/-36,506	65.2%	+/-0.1
Own children of the householder 6 to 17 years	47,007,147	+/-19,644	47,007,147	(X)
All parents in family in labor force	33,238,793	+/-49,036	70.7%	+/-0.1

Source: US Census Fact Finder, General Economic Characteristics, ACS 2017