Data Wrangling and Data Verbs

Code

Instructor - Soumya Mukherjee

Content Credit- Dr. Matthew Beckman and Olivia Beck

July 14, 2023

Agenda

- Introduce some software and commands that …
    - make it easy to access data tables and see how they are structured
        - For example: `data()`, `View()`, `help()`,
        - (more coming in Chapter 10)
    - learn about data verbs
    - implement two important data verbs: `group_by()` and `summarise()`

Three Important Concepts

1. Data can be usefully organized into tables with “cases” and “variables.”
    - In “tidy data” every case is the same sort of thing (e.g. a person, a car, a year, a country in a year)
    - We sometimes even modify data in order to change what the cases represent in order to better represent a point.
2. Data graphics and “glyph-ready” data
    - each case corresponds to a “glyph” (mark) on the graph
    - each variable to a graphical attribute of that glyph such as x- or y-position, color, size, length, shape, etc.
    - same is true for more technical tools (e.g., models, predictions, etc.)
3. When data are not yet in glyph-ready form, you can transform (i.e. wrangle) them into glyph-ready form.
    - Such transformations are accomplished by performing one or more of a small set of basic operations on data tables
    - This is the work of data “verbs”

Learning about the raw data

There are lots of ways to load data into your environment

- Most real data sources will require you to

    - read a file (e.g., CSV)
    - query a database (e.g., SQL)
    - configure an API
    - scrape from the web
- For convenience, many STAT 184 data sets are accessed from R packages or CSV files
- When acquiring data, it's very important to pause and think about data provenance/origins
    - What might be useful to learn?
    - How is this accomplished?
    - Why does it matter?

Recall: Key goals of a careful Exploratory Data Analysis?

1. **Examine the data source:** variable types, coding, missingness, summary statistics/plots, who/what/when/where/why/how data were collected
2. **Discover features that influence may modeling decisions:** investigate potential outliers, consideration for recoding variables (e.g., numeric data that's functionally dichotomous), evaluate correlation structure (e.g., autocorrelation, hierarchy, spatial/temporal proximity)
3. **Address research questions:** build intuition and note preliminary observations/conclusions related to each research question. Also, note observations that prompt you to refine your research questions or add new questions to investigate

A few simple commands to help us "Examine the data source":

- *Note:* often you need to examine information sources outside R to do a thorough examination.
- `help()` or `?` : if your data are part of an R package, this opens a help window with details about the data
- `data()` : if your data are part of an R package, this function loads the data set into your R environment and binds an object name
- `head(Dat)` : inspect the first few rows of `Dat`
- `View(Dat)` : opens a spreadsheet tab in RStudio showing `Dat` in it's entirety
    - You can also click on the table name in the "Enviornment" Pane
    - Bad form to call `View()` in the Rmd, use the console for this one.
    - `head()` is best in the Rmd

Guided practice

- `Minneapolis2013` data set in the `dcData` package
    - To do this, we need to download the package from GitHub

Hide

```
# Install the package from GitHub
# The very first time you run this, uncomment the 3 lines below

# install.packages("devtools")
# library(devtools)
# install_github("mdbeckman/dcData")
library(dcData)
data("Minneapolis2013", package = "dcData")
```

Discussion questions:

1. What is the setting for the data?
    - What are they about?
    - Who collected them?
    - Why were they collected?
    - etc
2. How many cases are there?
3. In your own words, what kind of thing do the cases represent?
4. How many variables are there? What are their names?
5. Pick out three of the variables and say whether
    - the variable is quantitative or categorical
    - if categorical (R calls this a “factor”), what are some levels of the variable
    - if quantitative, what are the units of measurement of the variable.

Click here (https://en.wikipedia.org/wiki/Instant-runoff_voting) to learn about rank choice voting (also called instant run off voting).

Additional practice

- `CatsUK` data
    - Tidy Tuesday (Jan 31, 2023)
    - URL: https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2023/2023-01-31/cats_uk_reference.csv (https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2023/2023-01-31/cats_uk_reference.csv)
- `HELPmiss`
    - from `mosaicData` package

Hide

```
# "Cat UK Reference" data from Tidy Tuesday--Jan 31, 2023

csv_path <- 'https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2023/2023-01-31/cats_uk_reference.cs
v'


CatsUK <- read_csv(file = csv_path)  # note the new function `read_csv()` from `tidyverse`
```

```
Rows: 101 Columns: 16── Column specification ────────────────────────────────
Delimiter: ","
chr  (6): tag_id, animal_id, animal_taxon, animal_re...
dbl  (4): prey_p_month, hrs_indoors, n_cats, age_years
lgl  (4): hunt, food_dry, food_wet, food_other
dttm (2): deploy_on_date, deploy_off_date
ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Hide

```
# HELP data
# install.packages("mosaicData")
library(mosaicData)

data("HELPmiss", package = "mosaicData")
```

Even more Datasets/DataFrames:

Data Frame	Source R Library
`HappyPlanetIndex`	`Lock5Data` library
`Minneapolis2013`	`dcData` library
`CountryData`	`dcData` library

Data Frame	Source R Library
`EmployedACS`	`Lock5Data` library
`Marriage`	`mosaicData` library

Discussion questions:

1. What is the setting for the data?
    - What are they about?
    - Who collected them?
    - Why were they collected?
    - etc
2. How many cases are there?
3. In your own words, what kind of thing do the cases represent?
4. How many variables are there? What are their names?
5. Pick out three of the variables and say whether
    - the variable is quantitative or categorical
    - if categorical (R calls this a “factor”), what are some levels of the variable
    - if quantitative, what are the units of measurement of the variable.

Why we wrangle

Consider the Minneapolis 2013 election data.

Hide

```
# Look at the first few rows of the dataframe
Minneapolis2013 %>%
  head()
```

	Precinct <chr>	First <chr>	Second <chr>	Third <chr>	Ward <chr>
1	P-10	BETSY HODGES	undervote	undervote	W-7
2	P-06	BOB FINE	MARK ANDREW	undervote	W-10

	Precinct <chr>	First <chr>	Second <chr>	Third <chr>	Ward <chr>
3	P-09	KURTIS W. HANNA	BOB FINE	MIKE GOULD	W-10
4	P-05	BETSY HODGES	DON SAMUELS	undervote	W-13
5	P-01	DON SAMUELS	undervote	undervote	W-5
6	P-04	undervote	undervote	undervote	W-6

6 rows

Hide

```
# Look at the last few rows of the dataframe
Minneapolis2013 %>%
  tail()
```

	Precinct <chr>	First <chr>	Second <chr>	Third <chr>	Ward <chr>
80096	P-01	BETSY HODGES	undervote	undervote	W-8
80097	P-06	BETSY HODGES	JACKIE CHERRYHOMES	MARK ANDREW	W-9
80098	P-02	MARK ANDREW	DON SAMUELS	DOUG MANN	W-10
80099	P-07	MARK ANDREW	BETSY HODGES	DON SAMUELS	W-8
80100	P-04	MARK ANDREW	BETSY HODGES	undervote	W-4
80101	P-09	MARK ANDREW	JEFFREY ALAN WAGNER	MIKE GOULD	W-13

6 rows

Hide

```
# No. of rows/cases
Minneapolis2013 %>%
  nrow()
```

```
[1] 80101
```

Hide

```
# Help documentation (Codebook) for the dataframe
help(Minneapolis2013)
```

Here's a bar chart that might be used to show the election results:

Hide

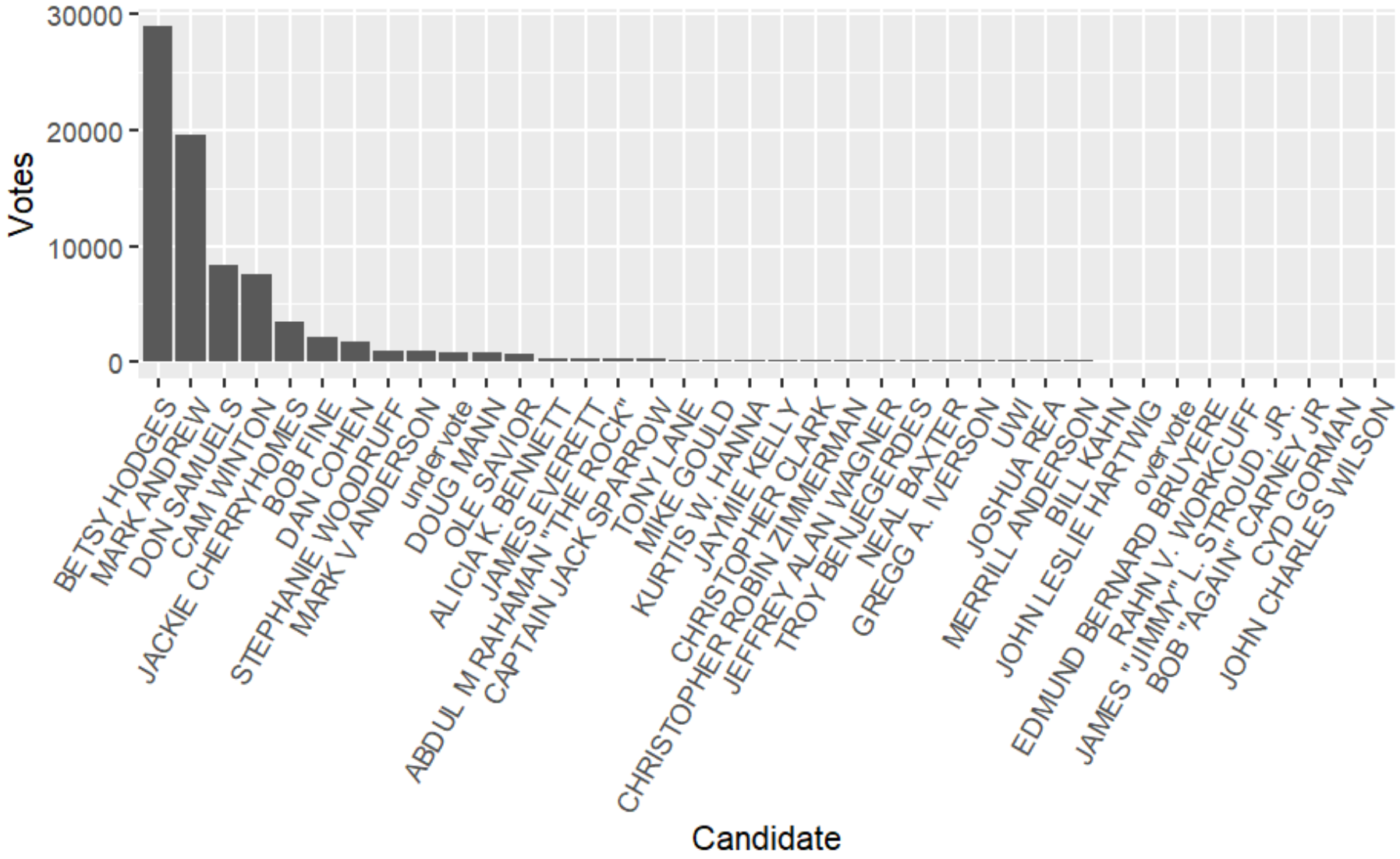
```
VoteResults <-
  Minneapolis2013 %>%
  group_by( First ) %>%
  summarise( votes = n() )

head(VoteResults)
```

First <chr>	**votes** <int>
ABDUL M RAHAMAN "THE ROCK"	338
ALICIA K. BENNETT	351
BETSY HODGES	28935
BILL KAHN	97
BOB "AGAIN" CARNEY JR	56
BOB FINE	2094

6 rows

Hide

```
# sorted bar chart (For the time being, create using esquisser from esquisse package)
ggplot(data = VoteResults,
       aes(x = reorder(First, desc(votes)), y = votes )) +
       geom_bar(stat = 'identity') +
       theme(axis.text.x = element_text(angle = 60, hjust = 1)) +
       ylab("Votes") +
       xlab("Candidate")
```



This graph reflects the following data table (only part of which is shown):

Hide

```
# we'll get to know these functions better soon
VoteResults %>%
  arrange( desc(votes) ) %>%
  head()
```

First <chr>	votes <int>
BETSY HODGES	28935
MARK ANDREW	19584
DON SAMUELS	8335
CAM WINTON	7511
JACKIE CHERRYHOMES	3524
BOB FINE	2094

6 rows

Compare the `Minneapolis2013` data table and the wrangled data table printed above.

1. Do they have the same number of cases?
2. Do the cases in the two tables represent the same sort of thing?
3. Do the two tables have any variable(s) in common?
4. How are the two tables are related to one another?

Why we wrangle

Data wrangling **prepares** the data for analysis.

- convert to tidy form for computing
- prepare glyph-ready data for visualization
- prepare data for modeling (e.g., exploratory, inferential, predictive)

Different types of functions

- Useful to have consistent language for data wrangling, just as we've done for visualization
- Some common function types:
    - **Reduction functions**
    - **Transformation functions**
    - **Data verbs**

For each type of function, what type of object is required as an input and what type of object is produced as a result?

- Relevant objects here include
    - scalars
    - variables
    - data frames

Different types of functions

- **Reduction functions**
    - inputs are **variables**
    - results are **scalar**
    - examples: `sum()`, `mean()`, `n()`
- **Transformation functions**
    - inputs are **variables**;
    - results are **variable**
    - examples: `weight / height`, `log10(population)`, `round(age)`
- **Data verbs**
    - inputs are **data frames**
    - results are **data frames**
    - examples: `summarise()`, `group_by()`

Any surprises above?

- `summarise()` as a data verb? Why not a reduction function??

Let’s use some other reduction functions, transformation functions, and data verbs with the some NFL data

Hide

```
dat.football <- read_tsv(file = "https://raw.githubusercontent.com/ada-lovecraft/ProcessingSketches/master/Bits%20and%20Pieces/Football_Stuff/data/nfl-salaries.tsv")
```

```
Rows: 1501 Columns: 6── Column specification ────────────────────────────────
Delimiter: "\t"
chr (3): PlayerName, Position, Team
dbl (3): Salary, NextSalary, SalaryCap
ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Hide

```
head(dat.football) #default is first 6 rows and all the columns
```

PlayerName <chr>	**Position** <chr>	**Salary** <dbl>	**NextSalary** <dbl>	**SalaryCap** <dbl>	**Team** <chr>
Tony Romo	QB	9000000	11500000	18905000	Dallas Cowboys
Anthony Spencer	LB	8800000	0	8800000	Dallas Cowboys
Jay Ratliff	DE	4875000	0	6475000	Dallas Cowboys
Terence Newman (buyout)	CB	4800000	0	4800000	Dallas Cowboys
Orlando Scandrick	CB	4700000	0	7700000	Dallas Cowboys
DeMarcus Ware	LB	4500000	5500000	10303000	Dallas Cowboys

6 rows

Hide

```
head(dat.football, n = 10)
```

PlayerName <chr>	Position <chr>	Salary <dbl>	NextSalary <dbl>	SalaryCap <dbl>	Team <chr>
Tony Romo	QB	9000000	11500000	18905000	Dallas Cowboys
Anthony Spencer	LB	8800000	0	8800000	Dallas Cowboys
Jay Ratliff	DE	4875000	0	6475000	Dallas Cowboys
Terence Newman (buyout)	CB	4800000	0	4800000	Dallas Cowboys
Orlando Scandrick	CB	4700000	0	7700000	Dallas Cowboys
DeMarcus Ware	LB	4500000	5500000	10303000	Dallas Cowboys
Jason Witten	TE	3641000	0	5841000	Dallas Cowboys
Marcus Spears	DE	2000000	2000000	2700000	Dallas Cowboys
Kenyon Coleman	DE	1900000	0	2245000	Dallas Cowboys
Jason Hatcher	DE	1500000	2000000	2100000	Dallas Cowboys

1-10 of 10 rows

Hide

```
dat.football %>%
  slice(1:10)
```

PlayerName <chr>	Position <chr>	Salary <dbl>	NextSalary <dbl>	SalaryCap <dbl>	Team <chr>
Tony Romo	QB	9000000	11500000	18905000	Dallas Cowboys
Anthony Spencer	LB	8800000	0	8800000	Dallas Cowboys
Jay Ratliff	DE	4875000	0	6475000	Dallas Cowboys

PlayerName <chr>	Position <chr>	Salary <dbl>	NextSalary <dbl>	SalaryCap <dbl>	Team <chr>
Terence Newman (buyout)	CB	4800000	0	4800000	Dallas Cowboys
Orlando Scandrick	CB	4700000	0	7700000	Dallas Cowboys
DeMarcus Ware	LB	4500000	5500000	10303000	Dallas Cowboys
Jason Witten	TE	3641000	0	5841000	Dallas Cowboys
Marcus Spears	DE	2000000	2000000	2700000	Dallas Cowboys
Kenyon Coleman	DE	1900000	0	2245000	Dallas Cowboys
Jason Hatcher	DE	1500000	2000000	2100000	Dallas Cowboys

1-10 of 10 rows

Hide

```
## Get the dimensions of the data
dim(dat.football)
```

```
[1] 1501    6
```

Hide

```
## Get the column names of the data
colnames(dat.football)
```

```
[1] "PlayerName" "Position"   "Salary"     "NextSalary"
[5] "SalaryCap"  "Team"
```

Hide

```
## Get the row names of the data
rownames(dat.football) #meaningless! (most times they will be)
```

```
  [1] "1"    "2"    "3"    "4"    "5"    "6"    "7"
  [8] "8"    "9"    "10"   "11"   "12"   "13"   "14"
 [15] "15"   "16"   "17"   "18"   "19"   "20"   "21"
 [22] "22"   "23"   "24"   "25"   "26"   "27"   "28"
 [29] "29"   "30"   "31"   "32"   "33"   "34"   "35"
 [36] "36"   "37"   "38"   "39"   "40"   "41"   "42"
 [43] "43"   "44"   "45"   "46"   "47"   "48"   "49"
 [50] "50"   "51"   "52"   "53"   "54"   "55"   "56"
 [57] "57"   "58"   "59"   "60"   "61"   "62"   "63"
 [64] "64"   "65"   "66"   "67"   "68"   "69"   "70"
 [71] "71"   "72"   "73"   "74"   "75"   "76"   "77"
 [78] "78"   "79"   "80"   "81"   "82"   "83"   "84"
 [85] "85"   "86"   "87"   "88"   "89"   "90"   "91"
 [92] "92"   "93"   "94"   "95"   "96"   "97"   "98"
 [99] "99"   "100"  "101"  "102"  "103"  "104"  "105"
[106] "106"  "107"  "108"  "109"  "110"  "111"  "112"
[113] "113"  "114"  "115"  "116"  "117"  "118"  "119"
[120] "120"  "121"  "122"  "123"  "124"  "125"  "126"
[127] "127"  "128"  "129"  "130"  "131"  "132"  "133"
[134] "134"  "135"  "136"  "137"  "138"  "139"  "140"
[141] "141"  "142"  "143"  "144"  "145"  "146"  "147"
[148] "148"  "149"  "150"  "151"  "152"  "153"  "154"
[155] "155"  "156"  "157"  "158"  "159"  "160"  "161"
[162] "162"  "163"  "164"  "165"  "166"  "167"  "168"
[169] "169"  "170"  "171"  "172"  "173"  "174"  "175"
[176] "176"  "177"  "178"  "179"  "180"  "181"  "182"
[183] "183"  "184"  "185"  "186"  "187"  "188"  "189"
[190] "190"  "191"  "192"  "193"  "194"  "195"  "196"
[197] "197"  "198"  "199"  "200"  "201"  "202"  "203"
[204] "204"  "205"  "206"  "207"  "208"  "209"  "210"
[211] "211"  "212"  "213"  "214"  "215"  "216"  "217"
[218] "218"  "219"  "220"  "221"  "222"  "223"  "224"
[225] "225"  "226"  "227"  "228"  "229"  "230"  "231"
[232] "232"  "233"  "234"  "235"  "236"  "237"  "238"
[239] "239"  "240"  "241"  "242"  "243"  "244"  "245"
[246] "246"  "247"  "248"  "249"  "250"  "251"  "252"
[253] "253"  "254"  "255"  "256"  "257"  "258"  "259"
[260] "260"  "261"  "262"  "263"  "264"  "265"  "266"
```

```
[267] "267"  "268"  "269"  "270"  "271"  "272"  "273"
[274] "274"  "275"  "276"  "277"  "278"  "279"  "280"
[281] "281"  "282"  "283"  "284"  "285"  "286"  "287"
[288] "288"  "289"  "290"  "291"  "292"  "293"  "294"
[295] "295"  "296"  "297"  "298"  "299"  "300"  "301"
[302] "302"  "303"  "304"  "305"  "306"  "307"  "308"
[309] "309"  "310"  "311"  "312"  "313"  "314"  "315"
[316] "316"  "317"  "318"  "319"  "320"  "321"  "322"
[323] "323"  "324"  "325"  "326"  "327"  "328"  "329"
[330] "330"  "331"  "332"  "333"  "334"  "335"  "336"
[337] "337"  "338"  "339"  "340"  "341"  "342"  "343"
[344] "344"  "345"  "346"  "347"  "348"  "349"  "350"
[351] "351"  "352"  "353"  "354"  "355"  "356"  "357"
[358] "358"  "359"  "360"  "361"  "362"  "363"  "364"
[365] "365"  "366"  "367"  "368"  "369"  "370"  "371"
[372] "372"  "373"  "374"  "375"  "376"  "377"  "378"
[379] "379"  "380"  "381"  "382"  "383"  "384"  "385"
[386] "386"  "387"  "388"  "389"  "390"  "391"  "392"
[393] "393"  "394"  "395"  "396"  "397"  "398"  "399"
[400] "400"  "401"  "402"  "403"  "404"  "405"  "406"
[407] "407"  "408"  "409"  "410"  "411"  "412"  "413"
[414] "414"  "415"  "416"  "417"  "418"  "419"  "420"
[421] "421"  "422"  "423"  "424"  "425"  "426"  "427"
[428] "428"  "429"  "430"  "431"  "432"  "433"  "434"
[435] "435"  "436"  "437"  "438"  "439"  "440"  "441"
[442] "442"  "443"  "444"  "445"  "446"  "447"  "448"
[449] "449"  "450"  "451"  "452"  "453"  "454"  "455"
[456] "456"  "457"  "458"  "459"  "460"  "461"  "462"
[463] "463"  "464"  "465"  "466"  "467"  "468"  "469"
[470] "470"  "471"  "472"  "473"  "474"  "475"  "476"
[477] "477"  "478"  "479"  "480"  "481"  "482"  "483"
[484] "484"  "485"  "486"  "487"  "488"  "489"  "490"
[491] "491"  "492"  "493"  "494"  "495"  "496"  "497"
[498] "498"  "499"  "500"  "501"  "502"  "503"  "504"
[505] "505"  "506"  "507"  "508"  "509"  "510"  "511"
[512] "512"  "513"  "514"  "515"  "516"  "517"  "518"
[519] "519"  "520"  "521"  "522"  "523"  "524"  "525"
[526] "526"  "527"  "528"  "529"  "530"  "531"  "532"
```

```
[533] "533"  "534"  "535"  "536"  "537"  "538"  "539"
[540] "540"  "541"  "542"  "543"  "544"  "545"  "546"
[547] "547"  "548"  "549"  "550"  "551"  "552"  "553"
[554] "554"  "555"  "556"  "557"  "558"  "559"  "560"
[561] "561"  "562"  "563"  "564"  "565"  "566"  "567"
[568] "568"  "569"  "570"  "571"  "572"  "573"  "574"
[575] "575"  "576"  "577"  "578"  "579"  "580"  "581"
[582] "582"  "583"  "584"  "585"  "586"  "587"  "588"
[589] "589"  "590"  "591"  "592"  "593"  "594"  "595"
[596] "596"  "597"  "598"  "599"  "600"  "601"  "602"
[603] "603"  "604"  "605"  "606"  "607"  "608"  "609"
[610] "610"  "611"  "612"  "613"  "614"  "615"  "616"
[617] "617"  "618"  "619"  "620"  "621"  "622"  "623"
[624] "624"  "625"  "626"  "627"  "628"  "629"  "630"
[631] "631"  "632"  "633"  "634"  "635"  "636"  "637"
[638] "638"  "639"  "640"  "641"  "642"  "643"  "644"
[645] "645"  "646"  "647"  "648"  "649"  "650"  "651"
[652] "652"  "653"  "654"  "655"  "656"  "657"  "658"
[659] "659"  "660"  "661"  "662"  "663"  "664"  "665"
[666] "666"  "667"  "668"  "669"  "670"  "671"  "672"
[673] "673"  "674"  "675"  "676"  "677"  "678"  "679"
[680] "680"  "681"  "682"  "683"  "684"  "685"  "686"
[687] "687"  "688"  "689"  "690"  "691"  "692"  "693"
[694] "694"  "695"  "696"  "697"  "698"  "699"  "700"
[701] "701"  "702"  "703"  "704"  "705"  "706"  "707"
[708] "708"  "709"  "710"  "711"  "712"  "713"  "714"
[715] "715"  "716"  "717"  "718"  "719"  "720"  "721"
[722] "722"  "723"  "724"  "725"  "726"  "727"  "728"
[729] "729"  "730"  "731"  "732"  "733"  "734"  "735"
[736] "736"  "737"  "738"  "739"  "740"  "741"  "742"
[743] "743"  "744"  "745"  "746"  "747"  "748"  "749"
[750] "750"  "751"  "752"  "753"  "754"  "755"  "756"
[757] "757"  "758"  "759"  "760"  "761"  "762"  "763"
[764] "764"  "765"  "766"  "767"  "768"  "769"  "770"
[771] "771"  "772"  "773"  "774"  "775"  "776"  "777"
[778] "778"  "779"  "780"  "781"  "782"  "783"  "784"
[785] "785"  "786"  "787"  "788"  "789"  "790"  "791"
[792] "792"  "793"  "794"  "795"  "796"  "797"  "798"
```

```
 [799] "799"  "800"  "801"  "802"  "803"  "804"  "805"
 [806] "806"  "807"  "808"  "809"  "810"  "811"  "812"
 [813] "813"  "814"  "815"  "816"  "817"  "818"  "819"
 [820] "820"  "821"  "822"  "823"  "824"  "825"  "826"
 [827] "827"  "828"  "829"  "830"  "831"  "832"  "833"
 [834] "834"  "835"  "836"  "837"  "838"  "839"  "840"
 [841] "841"  "842"  "843"  "844"  "845"  "846"  "847"
 [848] "848"  "849"  "850"  "851"  "852"  "853"  "854"
 [855] "855"  "856"  "857"  "858"  "859"  "860"  "861"
 [862] "862"  "863"  "864"  "865"  "866"  "867"  "868"
 [869] "869"  "870"  "871"  "872"  "873"  "874"  "875"
 [876] "876"  "877"  "878"  "879"  "880"  "881"  "882"
 [883] "883"  "884"  "885"  "886"  "887"  "888"  "889"
 [890] "890"  "891"  "892"  "893"  "894"  "895"  "896"
 [897] "897"  "898"  "899"  "900"  "901"  "902"  "903"
 [904] "904"  "905"  "906"  "907"  "908"  "909"  "910"
 [911] "911"  "912"  "913"  "914"  "915"  "916"  "917"
 [918] "918"  "919"  "920"  "921"  "922"  "923"  "924"
 [925] "925"  "926"  "927"  "928"  "929"  "930"  "931"
 [932] "932"  "933"  "934"  "935"  "936"  "937"  "938"
 [939] "939"  "940"  "941"  "942"  "943"  "944"  "945"
 [946] "946"  "947"  "948"  "949"  "950"  "951"  "952"
 [953] "953"  "954"  "955"  "956"  "957"  "958"  "959"
 [960] "960"  "961"  "962"  "963"  "964"  "965"  "966"
 [967] "967"  "968"  "969"  "970"  "971"  "972"  "973"
 [974] "974"  "975"  "976"  "977"  "978"  "979"  "980"
 [981] "981"  "982"  "983"  "984"  "985"  "986"  "987"
 [988] "988"  "989"  "990"  "991"  "992"  "993"  "994"
 [995] "995"  "996"  "997"  "998"  "999"  "1000"
 [ reached getOption("max.print") -- omitted 501 entries ]
```

Hide

```
## Get a summary of the data
## sumamry is not summarize!
summary(dat.football) # gives summary info by column
```

```
 PlayerName          Position
Length:1501        Length:1501
Class :character   Class :character
Mode  :character   Mode  :character



    Salary           NextSalary
Min.   :    2333   Min.   :       0
1st Qu.:  490000   1st Qu.:       0
Median :  615000   Median :  555000
Mean   : 1566829   Mean   : 1248008
3rd Qu.: 1700000   3rd Qu.:  900000
Max.   :18000000   Max.   :20000000
  SalaryCap            Team
Min.   :       0   Length:1501
1st Qu.:  515946   Class :character
Median :  770000   Mode  :character
Mean   : 2171577
3rd Qu.: 2700000
Max.   :20250000
```

Now lets look at some `tidyverse` functions.

Hide

```
#Filter
dat.football %>%
  filter(Team == "Denver Broncos")
```

PlayerName <chr>	Position <chr>	Salary <dbl>	NextSalary <dbl>	SalaryCap <dbl>	Team <chr>
Peyton Manning	QB	18000000	20000000	18000000	Denver Broncos
Elvis Dumervil	DE	14000000	12000000	14500000	Denver Broncos
Champ Bailey	CB	8000000	9000000	9500000	Denver Broncos

PlayerName <chr>	Position <chr>	Salary <dbl>	NextSalary <dbl>	SalaryCap <dbl>	Team <chr>
Brian Dawkins	S	6000000	6000000	9156000	Denver Broncos
D.J. Williams	LB	5000000	6000000	5000000	Denver Broncos
Andre' Goodman	CB	4620000	3960000	5580000	Denver Broncos
Ty Warren	DT	4000000	0	5250000	Denver Broncos
Chris Kuper	G	3500000	4500000	3500000	Denver Broncos
Ryan Clady	T	3500000	0	4010000	Denver Broncos
Matt Prater	K	2665000	0	2665000	Denver Broncos

1-10 of 58 rows

Previous **1** 2 3 4 5 6 Next

Hide

```
#Arrange
dat.football %>%
  arrange(Salary) #lowest to highest
```

PlayerName <chr>	Position <chr>	Salary <dbl>	NextSalary <dbl>	SalaryCap <dbl>	Team <chr>
Richard Dickson (buyout)	TE	2333	0	2333	Detroit Lions
Kevin Haslam (buyout)	T	3333	0	3333	Jacksonville Jaguars
Curtis Painter (buyout)	QB	22750	0	22750	Indianapolis Colts
Jon Corto (Buyout)	S	25000	0	25000	Buffalo Bills
George Selvie (buyout)	DE	27976	0	27976	St. Louis Rams
David Buehler (buyout)	K	37125	0	37125	Dallas Cowboys

PlayerName <chr>	Position <chr>	Salary <dbl>	NextSalary <dbl>	SalaryCap <dbl>	Team <chr>
Markell Carter	DE	70539	0	390000	New England Patriots
Morgan Trent (Buyout)	CB	84000	0	84000	Cincinnati Bengals
Anthony Herrera (buyout)	G	100000	0	100000	Minnesota Vikings
Jordan Todman (buyout)	RB	128094	0	128094	San Diego Chargers

1-10 of 1,501 rows Previous **1** 2 3 4 5 6 … 100 Next

Hide

```
dat.football %>%
  arrange(desc(Salary)) #highest to lowest
```

PlayerName <chr>	Position <chr>	Salary <dbl>	NextSalary <dbl>	SalaryCap <dbl>	Team <chr>
Peyton Manning	QB	18000000	20000000	18000000	Denver Broncos
Drew Brees	QB	15760000	0	15760000	New Orleans Saints
Dwight Freeney	DE	14035000	0	19035000	Indianapolis Colts
Elvis Dumervil	DE	14000000	12000000	14500000	Denver Broncos
Michael Vick	QB	12500000	12500000	13900000	Philadelphia Eagles
Sam Bradford	QB	12000000	9000000	15594800	St. Louis Rams
Jared Allen	DE	11619850	14280612	14203183	Minnesota Vikings
Matthew Stafford	QB	11500000	1200000	17258750	Detroit Lions
Matt Ryan	QB	11500000	10000000	13000000	Atlanta Falcons

PlayerName <chr>	Position <chr>	Salary <dbl>	NextSalary <dbl>	SalaryCap <dbl>	Team <chr>
Tamba Hali	DE	11250000	12250000	14250000	Kansas City Chiefs

1-10 of 1,501 rows Previous **1** 2 3 4 5 6 … 100 Next

Hide

```
#Select
dat.football %>%
  select(PlayerName, Position)
```

PlayerName <chr>	Position <chr>
Tony Romo	QB
Anthony Spencer	LB
Jay Ratliff	DE
Terence Newman (buyout)	CB
Orlando Scandrick	CB
DeMarcus Ware	LB
Jason Witten	TE
Marcus Spears	DE
Kenyon Coleman	DE
Jason Hatcher	DE

1-10 of 1,501 rows Previous **1** 2 3 4 5 6 … 100 Next

Hide

```
#Rename
dat.football %>%
  rename(TeamName = Team)
```

PlayerName <chr>	Position <chr>	Salary <dbl>	NextSalary <dbl>	SalaryCap <dbl>	TeamName <chr>
Tony Romo	QB	9000000	11500000	18905000	Dallas Cowboys
Anthony Spencer	LB	8800000	0	8800000	Dallas Cowboys
Jay Ratliff	DE	4875000	0	6475000	Dallas Cowboys
Terence Newman (buyout)	CB	4800000	0	4800000	Dallas Cowboys
Orlando Scandrick	CB	4700000	0	7700000	Dallas Cowboys
DeMarcus Ware	LB	4500000	5500000	10303000	Dallas Cowboys
Jason Witten	TE	3641000	0	5841000	Dallas Cowboys
Marcus Spears	DE	2000000	2000000	2700000	Dallas Cowboys
Kenyon Coleman	DE	1900000	0	2245000	Dallas Cowboys
Jason Hatcher	DE	1500000	2000000	2100000	Dallas Cowboys

Hide

```
#Mutate
dat.football %>%
  mutate(PercentOfCap = Salary / SalaryCap * 100)
```

PlayerName <chr>	Position <chr>	Salary <dbl>	NextSalary <dbl>	SalaryCap <dbl>	Team <chr>	PercentOfCap <dbl>
Tony Romo	QB	9000000	11500000	18905000	Dallas Cowboys	47.60645
Anthony Spencer	LB	8800000	0	8800000	Dallas Cowboys	100.00000
Jay Ratliff	DE	4875000	0	6475000	Dallas Cowboys	75.28958
Terence Newman (buyout)	CB	4800000	0	4800000	Dallas Cowboys	100.00000
Orlando Scandrick	CB	4700000	0	7700000	Dallas Cowboys	61.03896
DeMarcus Ware	LB	4500000	5500000	10303000	Dallas Cowboys	43.67660
Jason Witten	TE	3641000	0	5841000	Dallas Cowboys	62.33522
Marcus Spears	DE	2000000	2000000	2700000	Dallas Cowboys	74.07407
Kenyon Coleman	DE	1900000	0	2245000	Dallas Cowboys	84.63252
Jason Hatcher	DE	1500000	2000000	2100000	Dallas Cowboys	71.42857

Hide

```
#Group
dat.football %>%
  group_by(Team) #doesn't look like it did anything???
```

PlayerName <chr>	Position <chr>	Salary <dbl>	NextSalary <dbl>	SalaryCap <dbl>	Team <chr>
Tony Romo	QB	9000000	11500000	18905000	Dallas Cowboys
Anthony Spencer	LB	8800000	0	8800000	Dallas Cowboys
Jay Ratliff	DE	4875000	0	6475000	Dallas Cowboys
Terence Newman (buyout)	CB	4800000	0	4800000	Dallas Cowboys

PlayerName <chr>	Position <chr>	Salary <dbl>	NextSalary <dbl>	SalaryCap <dbl>	Team <chr>
Orlando Scandrick	CB	4700000	0	7700000	Dallas Cowboys
DeMarcus Ware	LB	4500000	5500000	10303000	Dallas Cowboys
Jason Witten	TE	3641000	0	5841000	Dallas Cowboys
Marcus Spears	DE	2000000	2000000	2700000	Dallas Cowboys
Kenyon Coleman	DE	1900000	0	2245000	Dallas Cowboys
Jason Hatcher	DE	1500000	2000000	2100000	Dallas Cowboys

1-10 of 1,501 rows

Previous **1** 2 3 4 5 6 … 100 Next

Hide

```
#Summarise
?summarise

dat.football %>%
  summarise(MeanSalary = mean(Salary))
```

MeanSalary <dbl>
1566829

1 row

Hide

```
dat.football %>%
  summarize(SdSalary = sd(Salary))
```

SdSalary <dbl>
2099740

1 row

Hide

```
dat.football %>%
  group_by(Team) %>%
  summarise(MeanSalary = mean(Salary), .groups = "keep" )
```

Team <chr>	MeanSalary <dbl>
Arizona Cardinals	1594186.0
Atlanta Falcons	1828406.9
Baltimore Ravens	2156606.1
Buffalo Bills	1315185.4
Carolina Panthers	1353845.5
Chicago Bears	1758005.6
Cincinnati Bengals	1283529.3
Cleveland Browns	1573352.4
Dallas Cowboys	1480814.0
Denver Broncos	1683837.3

1-10 of 31 rows Previous **1** 2 3 4 Next

Hide

```
NA
NA
```

Exploratory Analysis - Combining it all together

What is the highest salary?

Hide

```
max(dat.football$Salary)
```

```
[1] 1.8e+07
```

Which player has this salary?

Hide

```
# Method 1 (no tidyverse functions)
max.salary <- max(dat.football$Salary) #get the max salary
row.max.salary <- dat.football$Salary == max.salary
answer.1 <- dat.football$PlayerName[row.max.salary]

#c(1, 2, 3, 4)[c(FALSE, FALSE, TRUE, FALSE)]

# Method 2 (tidyverse functions)
answer.2 <- dat.football %>%
  filter(Salary == max(Salary) ) %>%
  select(PlayerName)

# Method 3 (tidyverse functions)
answer.3 <- dat.football %>%
  arrange(desc(Salary)) %>%
  slice(1) %>%
  select(PlayerName)

## Whats the benefit of using tidyverse functions?
library(utils)
object.size(c(max.salary, row.max.salary, answer.1))
```

```
12304 bytes
```

Hide

```
object.size(answer.2)
```

```
944 bytes
```

Hide

```
object.size(answer.3)
```

```
944 bytes
```

Hide

```
944/12304 # used only 7% of the storage space by using tidyverse!
```

```
[1] 0.07672302
```

What is the team with the highest paid roster, and what was their total pay? What is the team with the lowest paid roster, and what was their total pay?

Hide

```
Paid <- dat.football %>%
  group_by(Team)%>%
  summarize(PaidRoster = sum(Salary)) %>%
  arrange(desc(PaidRoster))

Paid[1, ] #highest paid
```

Team <chr>	PaidRoster <dbl>
Tampa Bay Buccaneers	106247707

1 row

Hide

```
# how many teams are in our data set>

dim(Paid)
```

```
[1] 31  2
```

Hide

```
length(unique(dat.football$Team))
```

```
[1] 31
```

Hide

```
Paid[31, ]
```

Team <chr>	PaidRoster <dbl>
Cincinnati Bengals	51341172

1 row

Hide

```
#Bonus Question, if I said this data was from 2016 what team is missing from our data?
sort(unique(dat.football$Team))
```

```
 [1] "Arizona Cardinals"    "Atlanta Falcons"
 [3] "Baltimore Ravens"     "Buffalo Bills"
 [5] "Carolina Panthers"    "Chicago Bears"
 [7] "Cincinnati Bengals"   "Cleveland Browns"
 [9] "Dallas Cowboys"       "Denver Broncos"
[11] "Detroit Lions"        "Green Bay Packers"
[13] "Houston Texans"       "Indianapolis Colts"
[15] "Jacksonville Jaguars" "Kansas City Chiefs"
[17] "Miami Dolphins"       "Minnesota Vikings"
[19] "New England Patriots" "New Orleans Saints"
[21] "New York Giants"      "New York Jets"
[23] "Oakland Raiders"      "Philadelphia Eagles"
[25] "Pittsburgh Steelers"  "San Diego Chargers"
[27] "San Francisco 49ers"  "Seattle Seahawks"
[29] "St. Louis Rams"       "Tampa Bay Buccaneers"
[31] "Washington Redskins"
```

Pivot wider and Pivot Longer

`pivot_wider()` and `pivot_longer()` are two VERY useful functions in the `tidyverse`. We do not need them this week, but I wanted to introduce them if you want to get ahead.

Hide

```
## Pivot Wider
?pivot_wider

# names_from = new column names
# value_from = values to fill in in the table
us_rent_income
```

GEOID <chr>	NAME <chr>	variable <chr>	estimate <dbl>	moe <dbl>
01	Alabama	income	24476	136
01	Alabama	rent	747	3
02	Alaska	income	32940	508
02	Alaska	rent	1200	13
04	Arizona	income	27517	148
04	Arizona	rent	972	4
05	Arkansas	income	23789	165
05	Arkansas	rent	709	5
06	California	income	29454	109
06	California	rent	1358	3

1-10 of 104 rows

Previous **1** 2 3 4 5 6 … 11 Next

Hide

```
us_rent_income %>%
  pivot_wider(
    names_from = variable,
    values_from = c(estimate, moe)
  )
```

GEOID <chr>	NAME <chr>	estimate_income <dbl>	estimate_rent <dbl>	moe_income <dbl>	moe_rent <dbl>
01	Alabama	24476	747	136	3
02	Alaska	32940	1200	508	13
04	Arizona	27517	972	148	4
05	Arkansas	23789	709	165	5
06	California	29454	1358	109	3
08	Colorado	32401	1125	109	5
09	Connecticut	35326	1123	195	5
10	Delaware	31560	1076	247	10
11	District of Columbia	43198	1424	681	17
12	Florida	25952	1077	70	3

Hide

```
# is the above table tidy? What is each case?

## Pivot Longer
?pivot_longer
#name_to = new column name that will contain the old column names
#values_to = new column name that will contain the data from the original table
relig_income
```

religion <chr>	<$10k <dbl>	$10-20k <dbl>	$20-30k <dbl>	$30-40k <dbl>	$40-50k <dbl>	$50-75k <dbl>	$75-100k <dbl>	$100-150k <dbl>	>150k <dbl>
Agnostic	27	34	60	81	76	137	122	109	84
Atheist	12	27	37	52	35	70	73	59	74
Buddhist	27	21	30	34	33	58	62	39	53
Catholic	418	617	732	670	638	1116	949	792	633
Don’t know/refused	15	14	15	11	10	35	21	17	18
Evangelical Prot	575	869	1064	982	881	1486	949	723	414
Hindu	1	9	7	9	11	34	47	48	54
Historically Black Prot	228	244	236	238	197	223	131	81	78
Jehovah's Witness	20	27	24	24	21	30	15	11	6
Jewish	19	19	25	25	30	95	69	87	151

Hide

```
relig_income %>%
  pivot_longer(!religion,  # every column but religion
               names_to = "income",
               values_to = "count")
```

religion <chr>	income <chr>	count <dbl>
Agnostic	<$10k	27
Agnostic	$10-20k	34
Agnostic	$20-30k	60
Agnostic	$30-40k	81

religion <chr>	income <chr>	count <dbl>
Agnostic	$40-50k	76
Agnostic	$50-75k	137
Agnostic	$75-100k	122
Agnostic	$100-150k	109
Agnostic	>150k	84
Agnostic	Don't know/refused	96

1-10 of 180 rows Previous **1** 2 3 4 5 6 … 18 Next

Hide

```
# Is the above table Tidy? What is a case?

world_bank_pop
```

country <chr>	indicator <chr>	2000 <dbl>	2001 <dbl>	2002 <dbl>	2003 <dbl>	2004 <dbl>
ABW	SP.URB.TOTL	4.162500e+04	4.202500e+04	4.219400e+04	4.227700e+04	4.231700e+04
ABW	SP.URB.GROW	1.664222e+00	9.563731e-01	4.013352e-01	1.965172e-01	9.456936e-02
ABW	SP.POP.TOTL	8.910100e+04	9.069100e+04	9.178100e+04	9.270100e+04	9.354000e+04
ABW	SP.POP.GROW	2.539234e+00	1.768757e+00	1.194718e+00	9.973955e-01	9.009892e-01
AFE	SP.URB.TOTL	1.155517e+08	1.197755e+08	1.242275e+08	1.288340e+08	1.336475e+08
AFE	SP.URB.GROW	3.602262e+00	3.655377e+00	3.716958e+00	3.708082e+00	3.736205e+00
AFE	SP.POP.TOTL	4.016006e+08	4.120019e+08	4.227411e+08	4.338075e+08	4.452816e+08
AFE	SP.POP.GROW	2.583579e+00	2.589961e+00	2.606598e+00	2.617764e+00	2.644968e+00

country <chr>	indicator <chr>	2000 <dbl>	2001 <dbl>	2002 <dbl>	2003 <dbl>	2004 <dbl>
AFG	SP.URB.TOTL	4.314700e+06	4.364773e+06	4.674867e+06	5.061866e+06	5.299549e+06
AFG	SP.URB.GROW	1.861377e+00	1.153839e+00	6.863453e+00	7.953448e+00	4.588653e+00

1-10 of 1,064 rows | 1-7 of 20 columns

Previous **1** 2 3 4 5 6 … 100 Next

Hide

```
world_bank_pop %>%
  pivot_longer(!c(country, indicator),
               names_to = "year",
               values_to = "count")
```

country <chr>	indicator <chr>	year <chr>	count <dbl>
ABW	SP.URB.TOTL	2000	4.162500e+04
ABW	SP.URB.TOTL	2001	4.202500e+04
ABW	SP.URB.TOTL	2002	4.219400e+04
ABW	SP.URB.TOTL	2003	4.227700e+04
ABW	SP.URB.TOTL	2004	4.231700e+04
ABW	SP.URB.TOTL	2005	4.239900e+04
ABW	SP.URB.TOTL	2006	4.255500e+04
ABW	SP.URB.TOTL	2007	4.272900e+04
ABW	SP.URB.TOTL	2008	4.290600e+04
ABW	SP.URB.TOTL	2009	4.307900e+04

1-10 of 19,152 rows

Previous **1** 2 3 4 5 6 … 100 Next

Hide

```
# is the above table tidy? What is a case?
```

Assignments before next lecture (July 17)

- Reading Quiz Chapter 7 (due Monday, July 17, 9:59 am)
- Activity: STAT184-HELPrct (Go through this over the weekend, will do this in class if needed, submit on Tuesday, July 18, 9:59am)