Week 10: Data Analysis 1 - Data Frames

EMSE 6574, Section 11

John Helveston

October 28, 2019

Quiz 4 - Strings!

20 minutes

- No calculators
- No notes
- No books
- No computers
- No phones

Announcements

1) Download the `week10notes.zip` file for class today (link in `slack/classroom`).

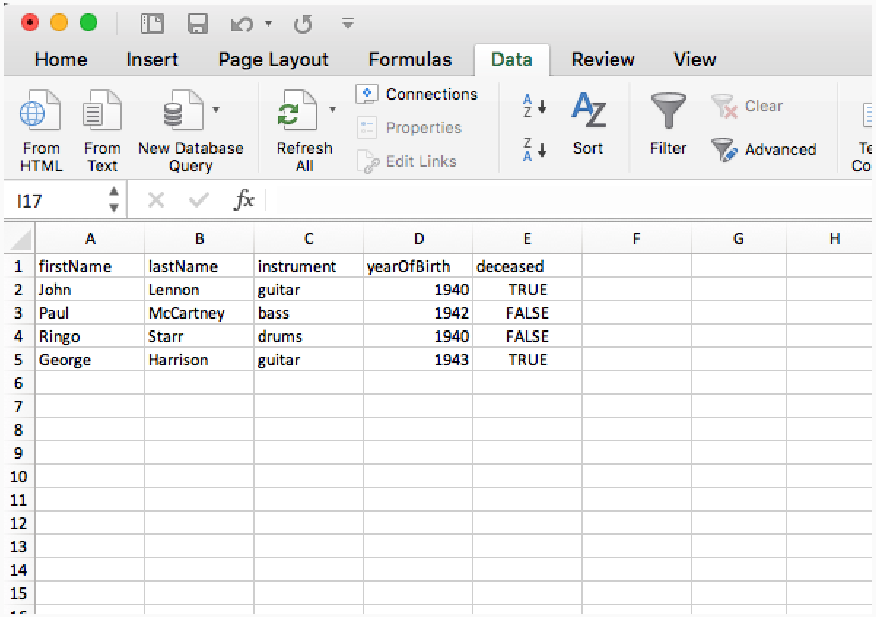
2) Make sure you have these packages installed and loaded:

```r
install.packages("stringr")
install.packages("dplyr")
install.packages("ggplot2")
install.packages("readr")
```

"The purpose of computing is insight, not numbers"

- Richard Hamming

The data frame...in Excel



	A	B	C	D	E
1	firstName	lastName	instrument	yearOfBirth	deceased
2	John	Lennon	guitar	1940	TRUE
3	Paul	McCartney	bass	1942	FALSE
4	Ringo	Starr	drums	1940	FALSE
5	George	Harrison	guitar	1943	TRUE

The data frame…in R

R:

```
## # A tibble: 4 x 5
##   firstName lastName  instrument yearOfBirth deceased
##   <chr>     <chr>     <chr>            <dbl> <lgl>
## 1 John      Lennon    guitar            1940 TRUE
## 2 Paul      McCartney bass              1942 FALSE
## 3 Ringo     Starr     drums             1940 FALSE
## 4 George    Harrison  guitar            1943 TRUE
```

Data frame columns are **vectors**

The **data frame** is a collection of **vectors** of the same length

```
beatles
```

```
## # A tibble: 4 x 5
##   firstName lastName  instrument yearOfBirth deceased
##   <chr>     <chr>     <chr>            <dbl> <lgl>
## 1 John      Lennon    guitar            1940 TRUE
## 2 Paul      McCartney bass              1942 FALSE
## 3 Ringo     Starr     drums             1940 FALSE
## 4 George    Harrison  guitar            1943 TRUE
```

You can access each vector (column) using the `$` symbol:

```
beatles$firstName
```

```
## [1] "John"   "Paul"   "Ringo"  "George"
```

```
beatles$lastName
```

```
## [1] "Lennon"    "McCartney" "Starr"     "Harrison"
```

Making a data frame with `tibble()`

```r
library(dplyr)
```

```r
beatles <- tibble(
    firstName   = c("John", "Paul", "Ringo", "George"),
    lastName    = c("Lennon", "McCartney", "Starr", "Harrison"),
    instrument  = c("guitar", "bass", "drums", "guitar"),
    yearOfBirth = c(1940, 1942, 1940, 1943),
    deceased    = c(TRUE, FALSE, FALSE, TRUE)
)
```

```r
beatles
```

```
## # A tibble: 4 x 5
##   firstName lastName  instrument yearOfBirth deceased
##   <chr>     <chr>     <chr>            <dbl> <lgl>
## 1 John      Lennon    guitar            1940 TRUE
## 2 Paul      McCartney bass              1942 FALSE
## 3 Ringo     Starr     drums             1940 FALSE
## 4 George    Harrison  guitar            1943 TRUE
```

Each vector must have the same length

```r
beatles <- tibble(
    firstName   = c("John", "Paul", "Ringo", "George", "BOB"),
    lastName    = c("Lennon", "McCartney", "Starr", "Harrison"),
    instrument  = c("guitar", "bass", "drums", "guitar"),
    yearOfBirth = c(1940, 1942, 1940, 1943),
    deceased    = c(TRUE, FALSE, FALSE, TRUE)
)
```

```
## Tibble columns must have consistent lengths, only values of length one are rec
## * Length 4: Columns `lastName`, `instrument`, `yearOfBirth`, `deceased`
## * Length 5: Column `firstName`
```

Data frame rows are **observations**

```
beatles
```

```
## # A tibble: 4 x 5
##   firstName lastName  instrument yearOfBirth deceased
##   <chr>     <chr>     <chr>            <dbl> <lgl>
## 1 John      Lennon    guitar            1940 TRUE
## 2 Paul      McCartney bass              1942 FALSE
## 3 Ringo     Starr     drums             1940 FALSE
## 4 George    Harrison  guitar            1943 TRUE
```

Data frame rows are **observations**

```
beatles
```

```
## # A tibble: 4 x 5
##   firstName lastName  instrument yearOfBirth deceased
##   <chr>     <chr>     <chr>            <dbl> <lgl>
## 1 John      Lennon    guitar            1940 TRUE
## 2 Paul      McCartney bass              1942 FALSE
## 3 Ringo     Starr     drums             1940 FALSE
## 4 George    Harrison  guitar            1943 TRUE
```

Example: Information about John Lennon is the first **row**

```
beatles[1,]
```

```
## # A tibble: 1 x 5
##   firstName lastName instrument yearOfBirth deceased
##   <chr>     <chr>    <chr>            <dbl> <lgl>
## 1 John      Lennon   guitar            1940 TRUE
```

Dimensions

```
## # A tibble: 4 x 5
##   firstName lastName  instrument yearOfBirth deceased
##   <chr>     <chr>     <chr>            <dbl> <lgl>
## 1 John      Lennon    guitar            1940 TRUE
## 2 Paul      McCartney bass              1942 FALSE
## 3 Ringo     Starr     drums             1940 FALSE
## 4 George    Harrison  guitar            1943 TRUE
```

```
nrow(beatles) # Number of rows
```

```
## [1] 4
```

```
ncol(beatles) # Number of columns
```

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

```
dim(beatles)  # Number of rows and columns
```

```
## [1] 4 5
```

Row and column names

Get the names of columns:

```
names(beatles)
```

```
## [1] "firstName"   "lastName"    "instrument"  "yearOfBirth" "deceased"
```

```
colnames(beatles)
```

```
## [1] "firstName"   "lastName"    "instrument"  "yearOfBirth" "deceased"
```

Get the names of rows:

```
rownames(beatles)
```

```
## [1] "1" "2" "3" "4"
```

Changing the column names

```
## # A tibble: 4 x 5
##   firstName lastName  instrument yearOfBirth deceased
##   <chr>     <chr>     <chr>            <dbl> <lgl>
## 1 John      Lennon    guitar            1940 TRUE
## 2 Paul      McCartney bass              1942 FALSE
## 3 Ringo     Starr     drums             1940 FALSE
## 4 George    Harrison  guitar            1943 TRUE
```

Change the column names:

```
colnames(beatles) <- c('one', 'two', 'three', 'four', 'five')
beatles
```

```
## # A tibble: 4 x 5
##   one    two       three   four five
##   <chr>  <chr>     <chr>  <dbl> <lgl>
## 1 John   Lennon    guitar  1940 TRUE
## 2 Paul   McCartney bass    1942 FALSE
## 3 Ringo  Starr     drums   1940 FALSE
## 4 George Harrison  guitar  1943 TRUE
```

Changing the column names

```
## # A tibble: 4 x 5
##   firstName lastName  instrument yearOfBirth deceased
##   <chr>     <chr>     <chr>            <dbl> <lgl>
## 1 John      Lennon    guitar            1940 TRUE
## 2 Paul      McCartney bass              1942 FALSE
## 3 Ringo     Starr     drums             1940 FALSE
## 4 George    Harrison  guitar            1943 TRUE
```

Change the column names:

```r
library(stringr)
colnames(beatles) <- str_to_upper(colnames(beatles))
beatles
```

```
## # A tibble: 4 x 5
##   FIRSTNAME LASTNAME  INSTRUMENT YEAROFBIRTH DECEASED
##   <chr>     <chr>     <chr>            <dbl> <lgl>
## 1 John      Lennon    guitar            1940 TRUE
## 2 Paul      McCartney bass              1942 FALSE
## 3 Ringo     Starr     drums             1940 FALSE
## 4 George    Harrison  guitar            1943 TRUE
```

Combining data frames

Combine by columns using `bind_cols()`:

```
names <- tibble(
    firstName = c("John", "Paul", "Ringo", "George"),
    lastName  = c("Lennon", "McCartney", "Starr", "Harrison")
)
instruments <- tibble(
    instrument = c("guitar", "bass", "drums", "guitar")
)
```

```
bind_cols(names, instruments)
```

```
## # A tibble: 4 x 3
##   firstName lastName  instrument
##   <chr>     <chr>     <chr>
## 1 John      Lennon    guitar
## 2 Paul      McCartney bass
## 3 Ringo     Starr     drums
## 4 George    Harrison  guitar
```

Combining data frames

Combine by rows using `bind_rows()`:

```r
members1 <- tibble(
    firstName = c("John", "Paul"),
    lastName  = c("Lennon", "McCartney")
)
members2 <- tibble(
    firstName = c("Ringo", "George"),
    lastName  = c("Starr", "Harrison")
)
```

```r
bind_rows(members1, members2)
```

```
## # A tibble: 4 x 2
##   firstName lastName
##   <chr>     <chr>
## 1 John      Lennon
## 2 Paul      McCartney
## 3 Ringo     Starr
## 4 George    Harrison
```

Combining data frames

Be careful - `bind_rows()` requires **exact same** columns names:

```
colnames(members2) <- c("firstName", "LastName")
bind_rows(members1, members2)
```

```
## # A tibble: 4 x 3
##   firstName lastName  LastName
##   <chr>     <chr>     <chr>
## 1 John      Lennon    <NA>
## 2 Paul      McCartney <NA>
## 3 Ringo     <NA>      Starr
## 4 George    <NA>      Harrison
```

Practice - Think, Pair, Share

```r
animals_farm = tibble(
    name           = c("cow", "horse"),
    sound          = c("moo", "neigh"),
    aveWeightLbs   = c(2400, 1500),
    aveLifeSpanYrs = c(20, 25)
)
animals_pet = tibble(
    name           = c("dog", "cat"),
    sound          = c("woof", "meow"),
    aveWeightLbs   = c(40, 8),
    aveLifeSpanYrs = c(10, 12)
)
```

Use R code to find answers to these questions:

1. How many rows are in the `animals_farm` data frame?
2. How many columns are in the `animals_pet` data frame?
3. Create a new data frame, `animals`, by combining `animals_farm` and `animals_pet`.
4. Create a new column in `animals` called `type` and set the values to `"farm"` or `"pet"`.
5. Change the column names of `animals` to title case.

Accessing elements

General form for indexing elements:

```
DF[ROWS, COLUMNS]
```

Select the element in row 1, column 2:

```
beatles[1, 2]
```

```
## # A tibble: 1 x 1
##   lastName
##   <chr>
## 1 Lennon
```

Select the elements in rows 1 & 2 and columns 2 & 3:

```
beatles[c(1, 2), c(2, 3)]
```

```
## # A tibble: 2 x 2
##   lastName  instrument
##   <chr>     <chr>
## 1 Lennon    guitar
## 2 McCartney bass
```

Accessing elements

Leaving row or column index blank means "selects all":

```
beatles[c(1, 2),]
```

```
## # A tibble: 2 x 5
##   firstName lastName  instrument yearOfBirth deceased
##   <chr>     <chr>     <chr>            <dbl> <lgl>
## 1 John      Lennon    guitar            1940 TRUE
## 2 Paul      McCartney bass              1942 FALSE
```

```
beatles[,c(1, 2)]
```

```
## # A tibble: 4 x 2
##   firstName lastName
##   <chr>     <chr>
## 1 John      Lennon
## 2 Paul      McCartney
## 3 Ringo     Starr
## 4 George    Harrison
```

Negative indices exclude row / column

Select all rows except the first:

```
beatles[-1, ]
```

```
## # A tibble: 3 x 5
##   firstName lastName  instrument yearOfBirth deceased
##   <chr>     <chr>     <chr>            <dbl> <lgl>
## 1 Paul      McCartney bass              1942 FALSE
## 2 Ringo     Starr     drums             1940 FALSE
## 3 George    Harrison  guitar            1943 TRUE
```

Select all columns except the first:

```
beatles[,-1]
```

```
## # A tibble: 4 x 4
##   lastName  instrument yearOfBirth deceased
##   <chr>     <chr>            <dbl> <lgl>
## 1 Lennon    guitar            1940 TRUE
## 2 McCartney bass              1942 FALSE
## 3 Starr     drums             1940 FALSE
## 4 Harrison  guitar            1943 TRUE
```

Using character indices

You can use a vector of column names to select columns:

```
beatles[,c('firstName', 'lastName')]
```

```
## # A tibble: 4 x 2
##   firstName lastName
##   <chr>     <chr>
## 1 John      Lennon
## 2 Paul      McCartney
## 3 Ringo     Starr
## 4 George    Harrison
```

Same thing, but just the first two rows:

```
beatles[1:2, c('firstName', 'lastName')]
```

```
## # A tibble: 2 x 2
##   firstName lastName
##   <chr>     <chr>
## 1 John      Lennon
## 2 Paul      McCartney
```

Use logical indices to filter rows

Example: What if want to filter rows to find which Beatles members were still alive?

First, create a logical vector using the `deceased` column:

```r
beatles$deceased == FALSE
```

```
## [1] FALSE  TRUE  TRUE FALSE
```

Next, insert this logical vector in the row position of `[,]`:

```r
beatles[beatles$deceased == FALSE,]
```

```
## # A tibble: 2 x 5
##   firstName lastName  instrument yearOfBirth deceased
##   <chr>     <chr>     <chr>            <dbl> <lgl>
## 1 Paul      McCartney bass              1942 FALSE
## 2 Ringo     Starr     drums             1940 FALSE
```

Creating new variables

Use the `$` symbol to create a new column

Add the hometown of the bandmembers:

```
beatles$hometown <- 'Liverpool'
beatles
```

```
## # A tibble: 4 x 6
##   firstName lastName  instrument yearOfBirth deceased hometown
##   <chr>     <chr>     <chr>            <dbl> <lgl>    <chr>
## 1 John      Lennon    guitar            1940 TRUE     Liverpool
## 2 Paul      McCartney bass              1942 FALSE    Liverpool
## 3 Ringo     Starr     drums             1940 FALSE    Liverpool
## 4 George    Harrison  guitar            1943 TRUE     Liverpool
```

Creating new variables

Use the `$` symbol to create a new column

Compute and add the age of the bandmembers:

```
beatles$age <- 2019 - beatles$yearOfBirth
beatles
```

```
## # A tibble: 4 x 7
##   firstName lastName  instrument yearOfBirth deceased hometown    age
##   <chr>     <chr>     <chr>            <dbl> <lgl>    <chr>     <dbl>
## 1 John      Lennon    guitar            1940 TRUE     Liverpool    79
## 2 Paul      McCartney bass              1942 FALSE    Liverpool    77
## 3 Ringo     Starr     drums             1940 FALSE    Liverpool    79
## 4 George    Harrison  guitar            1943 TRUE     Liverpool    76
```

Practice - Think, Pair, Share

```r
beatles <- tibble(
    firstName   = c("John", "Paul", "Ringo", "George"),
    lastName    = c("Lennon", "McCartney", "Starr", "Harrison"),
    instrument  = c("guitar", "bass", "drums", "guitar"),
    yearOfBirth = c(1940, 1942, 1940, 1943),
    deceased    = c(TRUE, FALSE, FALSE, TRUE)
)
```

Use R code to find answers to these questions:

1. Create a new column, `playsGuitar`, which is `TRUE` if the band member plays the guitar and `FALSE` otherwise.
2. Select the rows for the band members who have four-letter first names.
3. Create a new column, `fullName`, which contains the band member's first and last name separated by a space (e.g. `"John Lennon"`)

5 minute break - stand up, move around,

5 minutes

Getting the data from an R package

```r
install.packages("ggplot2")
library(ggplot2)
```

```r
data(package = "ggplot2")
```

Dataset	Description
diamonds	Prices of 50,000 round cut diamonds
economics	US economic time series
economics_long	US economic time series
faithfuld	2d density estimate of Old Faithful data
luv_colours	'colors()' in Luv space
midwest	Midwest demographics
mpg	Fuel economy data from 1999 and 2008 for 38 popular models of car
msleep	An updated and expanded version of the mammals sleep dataset
presidential	Terms of 11 presidents from Eisenhower to Obama
seals	Vector field of seal movements

Working with external datasets

Today's example: `msleep`

V. M. Savage and G. B. West. "A quantitative, theoretical framework for understanding mammalian sleep." *Proceedings of the National Academy of Sciences*, 104 (3):1051-1056, 2007.

Column Name	**Description**
name	Common name
genus	The taxonomic genus of animal
vore	Carnivore, omnivore or herbivore?
order	The taxonomic order of animal
conservation	The conservation status of the animal
sleep_total	Total amount of sleep, in hours
sleep_rem	REM sleep, in hours
sleep_cycle	Length of sleep cycle, in hours
awake	Amount of time spent awake, in hours
brainwt	Brain weight in kilograms
bodywt	Body weight in kilograms

Importing data from a file

Note the `msleep.csv` file in your `data` folder.

- **DO NOT** double-click it!
- **DO NOT** open it in Excel!

PSA: Excel **breaks** data

Import the .csv file:

```r
library(readr)
pathToData <- file.path('data', 'msleep.csv')
msleep <- read_csv(pathToData)
```

A note about file paths

When you open a `.Rproj` file, R sets your *working directory* to the location of that file.

To view your current *working directory*, use:

```
getwd()
```

```
## [1] "/Users/jhelvy/gh/2019-Fall/classNotes/10-dataframes"
```

The `file.path()` function creates a **local** path **from your working directory**

```
pathToData <- file.path('data', 'msleep.csv')
pathToData
```

```
## [1] "data/msleep.csv"
```

Avoid using **hard-coded** file paths, like this:

```
pathToData <- 'data/msleep.csv'
```

Previewing data frames: Dimensions

```r
nrow(msleep) # Number of rows
```

```
## [1] 83
```

```r
ncol(msleep) # Number of columns
```

```
## [1] 11
```

```r
dim(msleep)  # Number of rows and columns
```

```
## [1] 83 11
```

Previewing data frames: Content

Look at the data in a "spreadsheet"-like way:

```
View(msleep)
```

View the **first** 6 rows with `head()`, or **last** 6 rows with `tail()`:

```
head(msleep)
```

```
## # A tibble: 6 x 11
##   name  genus vore  order conservation sleep_total sleep_rem sleep_cycle
##   <chr> <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl>
## 1 Chee… Acin… carni Carn… lc                  12.1      NA        NA    
## 2 Owl … Aotus omni  Prim… <NA>                17         1.8      NA    
## 3 Moun… Aplo… herbi Rode… nt                  14.4       2.4      NA    
## 4 Grea… Blar… omni  Sori… lc                  14.9       2.3       0.133
## 5 Cow   Bos   herbi Arti… domesticated         4         0.7       0.667
## 6 Thre… Brad… herbi Pilo… <NA>                14.4       2.2       0.767
## # … with 3 more variables: awake <dbl>, brainwt <dbl>, bodywt <dbl>
```

Quick data summaries

Preview each variable with `str()` or `glimpse()`:

```
glimpse(msleep)
```

```
## Observations: 83
## Variables: 11
## $ name         <chr> "Cheetah", "Owl monkey", "Mountain beaver", "Greate…
## $ genus        <chr> "Acinonyx", "Aotus", "Aplodontia", "Blarina", "Bos"…
## $ vore         <chr> "carni", "omni", "herbi", "omni", "herbi", "herbi",…
## $ order        <chr> "Carnivora", "Primates", "Rodentia", "Soricomorpha"…
## $ conservation <chr> "lc", NA, "nt", "lc", "domesticated", NA, "vu", NA,…
## $ sleep_total  <dbl> 12.1, 17.0, 14.4, 14.9, 4.0, 14.4, 8.7, 7.0, 10.1, …
## $ sleep_rem    <dbl> NA, 1.8, 2.4, 2.3, 0.7, 2.2, 1.4, NA, 2.9, NA, 0.6,…
## $ sleep_cycle  <dbl> NA, NA, NA, 0.1333333, 0.6666667, 0.7666667, 0.3833…
## $ awake        <dbl> 11.9, 7.0, 9.6, 9.1, 20.0, 9.6, 15.3, 17.0, 13.9, 2…
## $ brainwt      <dbl> NA, 0.01550, NA, 0.00029, 0.42300, NA, NA, NA, 0.07…
## $ bodywt       <dbl> 50.000, 0.480, 1.350, 0.019, 600.000, 3.850, 20.490…
```

Group Practice

1) Use `read_csv()` and `file.path()` to load the `wildlife_impacts.csv` file that is in the `data` folder. Name the data frame object `df`.

2) Use the `df` object to answer the following questions:

- How many rows and columns are in the data frame?
- What type of data is each column?
- Preview the different columns - what do you think this data is about? What might one row represent?
- How many unique airports are in the data frame?
- What is the earliest and latest observation in the data frame?
- What is the lowest and highest cost of any one repair in the data frame?

Next week: better data wrangling with **dplyr**

Select rows with `filter()`

Example: Filter rows to find which Beatles members are still alive?

Base R:

```
beatles[beatles$deceased == FALSE,]
```

Select rows with `filter()`

Example: Filter rows to find which Beatles members are still alive?

Base R:

```r
beatles[beatles$deceased == FALSE,]
```

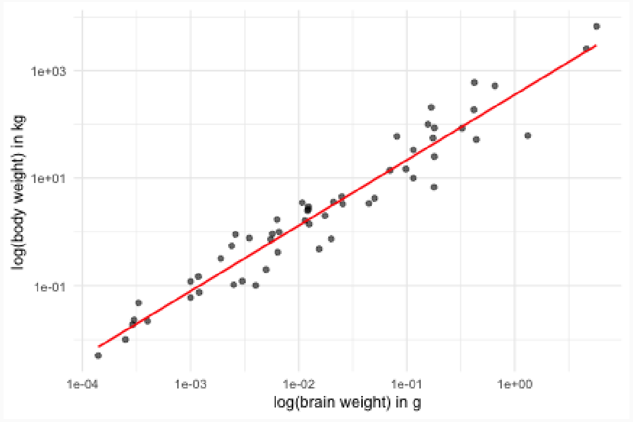
dplyr:

```r
filter(beatles, deceased == FALSE)
```

Next next week: plotting with **ggplot2**

Translating *data* into *insight*:

```r
library(ggplot2)
ggplot(msleep, aes(x=brainwt, y=bodywt)) +
    geom_point(alpha=0.6) +
    stat_smooth(method='lm', col='red', se=F, size=0.7) +
    scale_x_log10() +
    scale_y_log10() +
    labs(x='log(brain weight) in g', y='log(body weight) in kg') +
    theme_minimal()
```



A note about HW5

- You have what you need to start now.
- It will be *much* easier if you use the **dplyr** functions (i.e. read ahead).