R MarkDown

Josh Cohen

7/26/2021

## R Markdown Google Data Analytics Capstone Project (Using Mac)

This file details the the rationale behind various aspects of the R code utilized during the course of this project. I intend to use a conversational tone in my explanation of each section, operating under the assumption that if I can explain it in a way that anyone can understand, then I must have a mastery of the subject.

r = getOption("repos")  
r["CRAN"] = "http://cran.us.r-project.org"  
options(repos = r)

## 

## Setting The Stage With Packages

I began by installing all of the packages of which I am currently aware. Installing the packages is equivalent to downloading it to the system, but it doesn’t make them usable within the context of the program.

install.packages("tidyverse")

##   
## The downloaded binary packages are in  
## /var/folders/51/kzwrfpfd6s7gn8nyf6c6zr0w0000gn/T//Rtmpc0s489/downloaded\_packages

install.packages("lubridate")

##   
## The downloaded binary packages are in  
## /var/folders/51/kzwrfpfd6s7gn8nyf6c6zr0w0000gn/T//Rtmpc0s489/downloaded\_packages

install.packages("readxl")

##   
## The downloaded binary packages are in  
## /var/folders/51/kzwrfpfd6s7gn8nyf6c6zr0w0000gn/T//Rtmpc0s489/downloaded\_packages

install.packages("skimr")

##   
## The downloaded binary packages are in  
## /var/folders/51/kzwrfpfd6s7gn8nyf6c6zr0w0000gn/T//Rtmpc0s489/downloaded\_packages

install.packages("janitor")

##   
## The downloaded binary packages are in  
## /var/folders/51/kzwrfpfd6s7gn8nyf6c6zr0w0000gn/T//Rtmpc0s489/downloaded\_packages

install.packages("chron")

##   
## The downloaded binary packages are in  
## /var/folders/51/kzwrfpfd6s7gn8nyf6c6zr0w0000gn/T//Rtmpc0s489/downloaded\_packages

## 

## Making The Packages Discoverable/Usable

The library() function loads the packages into the program so that they are both discoverable and usable within RStudio.

library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.1 ──

## ✓ ggplot2 3.3.5 ✓ purrr 0.3.4  
## ✓ tibble 3.1.2 ✓ dplyr 1.0.7  
## ✓ tidyr 1.1.3 ✓ stringr 1.4.0  
## ✓ readr 1.4.0 ✓ forcats 0.5.1

## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(lubridate)

##   
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':  
##   
## date, intersect, setdiff, union

library(readxl)  
library(skimr)  
library(janitor)

##   
## Attaching package: 'janitor'

## The following objects are masked from 'package:stats':  
##   
## chisq.test, fisher.test

library(chron)

##   
## Attaching package: 'chron'

## The following objects are masked from 'package:lubridate':  
##   
## days, hours, minutes, seconds, years

## 

## Selecting The Working Directory

The working directory is like an address. R needs specific instructions to find the directory where the CSV or excel files are hidden before loading them into RStudio. Without identifying a working directory, it’s like trying to drive to a specific restaurant, which is an hour away from your home, without having an address, it’s fruitless. R doesn’t know what to do without a working directory and it returns an error. To alter the current working directory, you can apply the setwd() command and insert your desired working directory within the parentheses. You can find a file’s working directory by right clicking the file name in your Mac’s finder and selecting the “get info” option. The item listed next to the word “where” is the working directory.

## &

## Read In and Rename The CSV Files

Using the read\_csv() command loads csv files into RStudio. Sometimes it helps to rename the files in a way that makes them easier to identify and it also simplifies the process of including them in future formulas. Dropping in one of these assignment operators “<-” assigns a new name to a whole file or a column that has been manipulated.

setwd("/Users/cjmai/Documents/Google Data Analytics Bike Share Project 1/CSV")  
  
q2\_2019 <- read\_csv("Divvy\_Trips\_2019\_Q2")

##   
## ── Column specification ────────────────────────────────────────────────────────  
## cols(  
## `01 - Rental Details Rental ID` = col\_double(),  
## `01 - Rental Details Local Start Time` = col\_datetime(format = ""),  
## `01 - Rental Details Local End Time` = col\_datetime(format = ""),  
## `01 - Rental Details Bike ID` = col\_double(),  
## `01 - Rental Details Duration In Seconds Uncapped` = col\_number(),  
## `03 - Rental Start Station ID` = col\_double(),  
## `03 - Rental Start Station Name` = col\_character(),  
## `02 - Rental End Station ID` = col\_double(),  
## `02 - Rental End Station Name` = col\_character(),  
## `User Type` = col\_character(),  
## `Member Gender` = col\_character(),  
## `05 - Member Details Member Birthday Year` = col\_double()  
## )

q3\_2019 <- read\_csv("Divvy\_Trips\_2019\_Q3.csv")

##   
## ── Column specification ────────────────────────────────────────────────────────  
## cols(  
## trip\_id = col\_double(),  
## start\_time = col\_datetime(format = ""),  
## end\_time = col\_datetime(format = ""),  
## bikeid = col\_double(),  
## tripduration = col\_number(),  
## from\_station\_id = col\_double(),  
## from\_station\_name = col\_character(),  
## to\_station\_id = col\_double(),  
## to\_station\_name = col\_character(),  
## usertype = col\_character(),  
## gender = col\_character(),  
## birthyear = col\_double()  
## )

q4\_2019 <- read\_csv("Divvy\_Trips\_2019\_Q4.csv")

##   
## ── Column specification ────────────────────────────────────────────────────────  
## cols(  
## trip\_id = col\_double(),  
## start\_time = col\_datetime(format = ""),  
## end\_time = col\_datetime(format = ""),  
## bikeid = col\_double(),  
## tripduration = col\_number(),  
## from\_station\_id = col\_double(),  
## from\_station\_name = col\_character(),  
## to\_station\_id = col\_double(),  
## to\_station\_name = col\_character(),  
## usertype = col\_character(),  
## gender = col\_character(),  
## birthyear = col\_double()  
## )

q1\_2020 <- read\_csv("Divvy\_Trips\_2020\_Q1.csv")

##   
## ── Column specification ────────────────────────────────────────────────────────  
## cols(  
## ride\_id = col\_character(),  
## rideable\_type = col\_character(),  
## started\_at = col\_datetime(format = ""),  
## ended\_at = col\_datetime(format = ""),  
## start\_station\_name = col\_character(),  
## start\_station\_id = col\_double(),  
## end\_station\_name = col\_character(),  
## end\_station\_id = col\_double(),  
## start\_lat = col\_double(),  
## start\_lng = col\_double(),  
## end\_lat = col\_double(),  
## end\_lng = col\_double(),  
## member\_casual = col\_character()  
## )

## 

## Renaming Columns

Our end goal is to join the 4 DataFrames into a single DataFrame. Combining the DataFrames requires us to homogenize the column names so that they can find their equivalent column in the other DataFrames and pair up, which is what the rename() function is used for. We put the DataFrame that we are using before the first comma, next we list the new column names paired with the old column names, separated by an equals sign. This resets the old column names to the new column names.

(q4\_2019 <- rename(q4\_2019  
 ,ride\_id = trip\_id  
 ,rideable\_type = bikeid   
 ,started\_at = start\_time   
 ,ended\_at = end\_time   
 ,start\_station\_name = from\_station\_name   
 ,start\_station\_id = from\_station\_id   
 ,end\_station\_name = to\_station\_name   
 ,end\_station\_id = to\_station\_id   
 ,member\_casual = usertype))

## # A tibble: 704,054 x 12  
## ride\_id started\_at ended\_at rideable\_type tripduration  
## <dbl> <dttm> <dttm> <dbl> <dbl>  
## 1 25223640 2019-10-01 00:01:39 2019-10-01 00:17:20 2215 940  
## 2 25223641 2019-10-01 00:02:16 2019-10-01 00:06:34 6328 258  
## 3 25223642 2019-10-01 00:04:32 2019-10-01 00:18:43 3003 850  
## 4 25223643 2019-10-01 00:04:32 2019-10-01 00:43:43 3275 2350  
## 5 25223644 2019-10-01 00:04:34 2019-10-01 00:35:42 5294 1867  
## 6 25223645 2019-10-01 00:04:38 2019-10-01 00:10:51 1891 373  
## 7 25223646 2019-10-01 00:04:52 2019-10-01 00:22:45 1061 1072  
## 8 25223647 2019-10-01 00:04:57 2019-10-01 00:29:16 1274 1458  
## 9 25223648 2019-10-01 00:05:20 2019-10-01 00:29:18 6011 1437  
## 10 25223649 2019-10-01 00:05:20 2019-10-01 02:23:46 2957 8306  
## # … with 704,044 more rows, and 7 more variables: start\_station\_id <dbl>,  
## # start\_station\_name <chr>, end\_station\_id <dbl>, end\_station\_name <chr>,  
## # member\_casual <chr>, gender <chr>, birthyear <dbl>

(q3\_2019 <- rename(q3\_2019  
 ,ride\_id = trip\_id  
 ,rideable\_type = bikeid   
 ,started\_at = start\_time   
 ,ended\_at = end\_time   
 ,start\_station\_name = from\_station\_name   
 ,start\_station\_id = from\_station\_id   
 ,end\_station\_name = to\_station\_name   
 ,end\_station\_id = to\_station\_id   
 ,member\_casual = usertype))

## # A tibble: 1,640,718 x 12  
## ride\_id started\_at ended\_at rideable\_type tripduration  
## <dbl> <dttm> <dttm> <dbl> <dbl>  
## 1 23479388 2019-07-01 00:00:27 2019-07-01 00:20:41 3591 1214  
## 2 23479389 2019-07-01 00:01:16 2019-07-01 00:18:44 5353 1048  
## 3 23479390 2019-07-01 00:01:48 2019-07-01 00:27:42 6180 1554  
## 4 23479391 2019-07-01 00:02:07 2019-07-01 00:27:10 5540 1503  
## 5 23479392 2019-07-01 00:02:13 2019-07-01 00:22:26 6014 1213  
## 6 23479393 2019-07-01 00:02:21 2019-07-01 00:07:31 4941 310  
## 7 23479394 2019-07-01 00:02:24 2019-07-01 00:23:12 3770 1248  
## 8 23479395 2019-07-01 00:02:26 2019-07-01 00:28:16 5442 1550  
## 9 23479396 2019-07-01 00:02:34 2019-07-01 00:28:57 2957 1583  
## 10 23479397 2019-07-01 00:02:45 2019-07-01 00:29:14 6091 1589  
## # … with 1,640,708 more rows, and 7 more variables: start\_station\_id <dbl>,  
## # start\_station\_name <chr>, end\_station\_id <dbl>, end\_station\_name <chr>,  
## # member\_casual <chr>, gender <chr>, birthyear <dbl>

(q2\_2019 <- rename(q2\_2019  
 ,ride\_id = "01 - Rental Details Rental ID"  
 ,rideable\_type = "01 - Rental Details Bike ID"   
 ,started\_at = "01 - Rental Details Local Start Time"   
 ,ended\_at = "01 - Rental Details Local End Time"   
 ,start\_station\_name = "03 - Rental Start Station Name"   
 ,start\_station\_id = "03 - Rental Start Station ID"  
 ,end\_station\_name = "02 - Rental End Station Name"   
 ,end\_station\_id = "02 - Rental End Station ID"  
 ,member\_casual = "User Type"))

## # A tibble: 1,108,163 x 12  
## ride\_id started\_at ended\_at rideable\_type  
## <dbl> <dttm> <dttm> <dbl>  
## 1 22178529 2019-04-01 00:02:22 2019-04-01 00:09:48 6251  
## 2 22178530 2019-04-01 00:03:02 2019-04-01 00:20:30 6226  
## 3 22178531 2019-04-01 00:11:07 2019-04-01 00:15:19 5649  
## 4 22178532 2019-04-01 00:13:01 2019-04-01 00:18:58 4151  
## 5 22178533 2019-04-01 00:19:26 2019-04-01 00:36:13 3270  
## 6 22178534 2019-04-01 00:19:39 2019-04-01 00:23:56 3123  
## 7 22178535 2019-04-01 00:26:33 2019-04-01 00:35:41 6418  
## 8 22178536 2019-04-01 00:29:48 2019-04-01 00:36:11 4513  
## 9 22178537 2019-04-01 00:32:07 2019-04-01 01:07:44 3280  
## 10 22178538 2019-04-01 00:32:19 2019-04-01 01:07:39 5534  
## # … with 1,108,153 more rows, and 8 more variables:  
## # 01 - Rental Details Duration In Seconds Uncapped <dbl>,  
## # start\_station\_id <dbl>, start\_station\_name <chr>, end\_station\_id <dbl>,  
## # end\_station\_name <chr>, member\_casual <chr>, Member Gender <chr>,  
## # 05 - Member Details Member Birthday Year <dbl>

## Structure

This gives the structure of each DataFrame.

str(q1\_2020)

## spec\_tbl\_df [426,887 × 13] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)  
## $ ride\_id : chr [1:426887] "EACB19130B0CDA4A" "8FED874C809DC021" "789F3C21E472CA96" "C9A388DAC6ABF313" ...  
## $ rideable\_type : chr [1:426887] "docked\_bike" "docked\_bike" "docked\_bike" "docked\_bike" ...  
## $ started\_at : POSIXct[1:426887], format: "2020-01-21 20:06:59" "2020-01-30 14:22:39" ...  
## $ ended\_at : POSIXct[1:426887], format: "2020-01-21 20:14:30" "2020-01-30 14:26:22" ...  
## $ start\_station\_name: chr [1:426887] "Western Ave & Leland Ave" "Clark St & Montrose Ave" "Broadway & Belmont Ave" "Clark St & Randolph St" ...  
## $ start\_station\_id : num [1:426887] 239 234 296 51 66 212 96 96 212 38 ...  
## $ end\_station\_name : chr [1:426887] "Clark St & Leland Ave" "Southport Ave & Irving Park Rd" "Wilton Ave & Belmont Ave" "Fairbanks Ct & Grand Ave" ...  
## $ end\_station\_id : num [1:426887] 326 318 117 24 212 96 212 212 96 100 ...  
## $ start\_lat : num [1:426887] 42 42 41.9 41.9 41.9 ...  
## $ start\_lng : num [1:426887] -87.7 -87.7 -87.6 -87.6 -87.6 ...  
## $ end\_lat : num [1:426887] 42 42 41.9 41.9 41.9 ...  
## $ end\_lng : num [1:426887] -87.7 -87.7 -87.7 -87.6 -87.6 ...  
## $ member\_casual : chr [1:426887] "member" "member" "member" "member" ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. ride\_id = col\_character(),  
## .. rideable\_type = col\_character(),  
## .. started\_at = col\_datetime(format = ""),  
## .. ended\_at = col\_datetime(format = ""),  
## .. start\_station\_name = col\_character(),  
## .. start\_station\_id = col\_double(),  
## .. end\_station\_name = col\_character(),  
## .. end\_station\_id = col\_double(),  
## .. start\_lat = col\_double(),  
## .. start\_lng = col\_double(),  
## .. end\_lat = col\_double(),  
## .. end\_lng = col\_double(),  
## .. member\_casual = col\_character()  
## .. )

str(q4\_2019)

## spec\_tbl\_df [704,054 × 12] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)  
## $ ride\_id : num [1:704054] 25223640 25223641 25223642 25223643 25223644 ...  
## $ started\_at : POSIXct[1:704054], format: "2019-10-01 00:01:39" "2019-10-01 00:02:16" ...  
## $ ended\_at : POSIXct[1:704054], format: "2019-10-01 00:17:20" "2019-10-01 00:06:34" ...  
## $ rideable\_type : num [1:704054] 2215 6328 3003 3275 5294 ...  
## $ tripduration : num [1:704054] 940 258 850 2350 1867 ...  
## $ start\_station\_id : num [1:704054] 20 19 84 313 210 156 84 156 156 336 ...  
## $ start\_station\_name: chr [1:704054] "Sheffield Ave & Kingsbury St" "Throop (Loomis) St & Taylor St" "Milwaukee Ave & Grand Ave" "Lakeview Ave & Fullerton Pkwy" ...  
## $ end\_station\_id : num [1:704054] 309 241 199 290 382 226 142 463 463 336 ...  
## $ end\_station\_name : chr [1:704054] "Leavitt St & Armitage Ave" "Morgan St & Polk St" "Wabash Ave & Grand Ave" "Kedzie Ave & Palmer Ct" ...  
## $ member\_casual : chr [1:704054] "Subscriber" "Subscriber" "Subscriber" "Subscriber" ...  
## $ gender : chr [1:704054] "Male" "Male" "Female" "Male" ...  
## $ birthyear : num [1:704054] 1987 1998 1991 1990 1987 ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. trip\_id = col\_double(),  
## .. start\_time = col\_datetime(format = ""),  
## .. end\_time = col\_datetime(format = ""),  
## .. bikeid = col\_double(),  
## .. tripduration = col\_number(),  
## .. from\_station\_id = col\_double(),  
## .. from\_station\_name = col\_character(),  
## .. to\_station\_id = col\_double(),  
## .. to\_station\_name = col\_character(),  
## .. usertype = col\_character(),  
## .. gender = col\_character(),  
## .. birthyear = col\_double()  
## .. )

str(q3\_2019)

## spec\_tbl\_df [1,640,718 × 12] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)  
## $ ride\_id : num [1:1640718] 23479388 23479389 23479390 23479391 23479392 ...  
## $ started\_at : POSIXct[1:1640718], format: "2019-07-01 00:00:27" "2019-07-01 00:01:16" ...  
## $ ended\_at : POSIXct[1:1640718], format: "2019-07-01 00:20:41" "2019-07-01 00:18:44" ...  
## $ rideable\_type : num [1:1640718] 3591 5353 6180 5540 6014 ...  
## $ tripduration : num [1:1640718] 1214 1048 1554 1503 1213 ...  
## $ start\_station\_id : num [1:1640718] 117 381 313 313 168 300 168 313 43 43 ...  
## $ start\_station\_name: chr [1:1640718] "Wilton Ave & Belmont Ave" "Western Ave & Monroe St" "Lakeview Ave & Fullerton Pkwy" "Lakeview Ave & Fullerton Pkwy" ...  
## $ end\_station\_id : num [1:1640718] 497 203 144 144 62 232 62 144 195 195 ...  
## $ end\_station\_name : chr [1:1640718] "Kimball Ave & Belmont Ave" "Western Ave & 21st St" "Larrabee St & Webster Ave" "Larrabee St & Webster Ave" ...  
## $ member\_casual : chr [1:1640718] "Subscriber" "Customer" "Customer" "Customer" ...  
## $ gender : chr [1:1640718] "Male" NA NA NA ...  
## $ birthyear : num [1:1640718] 1992 NA NA NA NA ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. trip\_id = col\_double(),  
## .. start\_time = col\_datetime(format = ""),  
## .. end\_time = col\_datetime(format = ""),  
## .. bikeid = col\_double(),  
## .. tripduration = col\_number(),  
## .. from\_station\_id = col\_double(),  
## .. from\_station\_name = col\_character(),  
## .. to\_station\_id = col\_double(),  
## .. to\_station\_name = col\_character(),  
## .. usertype = col\_character(),  
## .. gender = col\_character(),  
## .. birthyear = col\_double()  
## .. )

str(q2\_2019)

## spec\_tbl\_df [1,108,163 × 12] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)  
## $ ride\_id : num [1:1108163] 22178529 22178530 22178531 22178532 22178533 ...  
## $ started\_at : POSIXct[1:1108163], format: "2019-04-01 00:02:22" "2019-04-01 00:03:02" ...  
## $ ended\_at : POSIXct[1:1108163], format: "2019-04-01 00:09:48" "2019-04-01 00:20:30" ...  
## $ rideable\_type : num [1:1108163] 6251 6226 5649 4151 3270 ...  
## $ 01 - Rental Details Duration In Seconds Uncapped: num [1:1108163] 446 1048 252 357 1007 ...  
## $ start\_station\_id : num [1:1108163] 81 317 283 26 202 420 503 260 211 211 ...  
## $ start\_station\_name : chr [1:1108163] "Daley Center Plaza" "Wood St & Taylor St" "LaSalle St & Jackson Blvd" "McClurg Ct & Illinois St" ...  
## $ end\_station\_id : num [1:1108163] 56 59 174 133 129 426 500 499 211 211 ...  
## $ end\_station\_name : chr [1:1108163] "Desplaines St & Kinzie St" "Wabash Ave & Roosevelt Rd" "Canal St & Madison St" "Kingsbury St & Kinzie St" ...  
## $ member\_casual : chr [1:1108163] "Subscriber" "Subscriber" "Subscriber" "Subscriber" ...  
## $ Member Gender : chr [1:1108163] "Male" "Female" "Male" "Male" ...  
## $ 05 - Member Details Member Birthday Year : num [1:1108163] 1975 1984 1990 1993 1992 ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. `01 - Rental Details Rental ID` = col\_double(),  
## .. `01 - Rental Details Local Start Time` = col\_datetime(format = ""),  
## .. `01 - Rental Details Local End Time` = col\_datetime(format = ""),  
## .. `01 - Rental Details Bike ID` = col\_double(),  
## .. `01 - Rental Details Duration In Seconds Uncapped` = col\_number(),  
## .. `03 - Rental Start Station ID` = col\_double(),  
## .. `03 - Rental Start Station Name` = col\_character(),  
## .. `02 - Rental End Station ID` = col\_double(),  
## .. `02 - Rental End Station Name` = col\_character(),  
## .. `User Type` = col\_character(),  
## .. `Member Gender` = col\_character(),  
## .. `05 - Member Details Member Birthday Year` = col\_double()  
## .. )

## 

## Changing The Columns Data Type

Mutate is an interesting function that can serve a variety of purposes. Mutate can change aspects of a column within a DataFrame, add a column to a DataFrame, etc.

q4\_2019 <- mutate(q4\_2019, ride\_id = as.character(ride\_id)  
 ,rideable\_type = as.character(rideable\_type))   
q3\_2019 <- mutate(q3\_2019, ride\_id = as.character(ride\_id)  
 ,rideable\_type = as.character(rideable\_type))   
q2\_2019 <- mutate(q2\_2019, ride\_id = as.character(ride\_id)  
 ,rideable\_type = as.character(rideable\_type))

## 

## Merging DataFrames

Once the column names and data types are matched up across the various DataFrames they can be stacked to form a new DataFrame. The bind\_rows() function will combine all of the DataFrames listed within the parenthesis. Using the “<-” operator will allow you to assign a different name to your newly created aggregate DataFrame.

all\_trips <- bind\_rows(q2\_2019, q3\_2019, q4\_2019, q1\_2020)

## 

## Dropping Irrelevant Columns

Not every column in a DataFrame will be useful in relation to helping to answer the question at hand. Some columns in certain DataFrames might have no equivalent in other DataFrames. Whatever the reason, there is always a chance that certain columns shouldn’t find a home in your new DataFrame. This formula selects certain columns and subtracts them from the DataFrame, hence “select” and “- or subtract” and “c for column”, so this makes a lot of sense.

all\_trips <- all\_trips %>%   
 select(-c(start\_lat, start\_lng, end\_lat, end\_lng, birthyear, gender, "01 - Rental Details Duration In Seconds Uncapped", "05 - Member Details Member Birthday Year", "Member Gender", "tripduration"))

## 

## Overview

The following is a set of functions that give an overview of the combined DataFrame, therefore, we know what we are dealing with before we proceed.

colnames(all\_trips)

## [1] "ride\_id" "started\_at" "ended\_at"   
## [4] "rideable\_type" "start\_station\_id" "start\_station\_name"  
## [7] "end\_station\_id" "end\_station\_name" "member\_casual"

nrow(all\_trips)

## [1] 3879822

dim(all\_trips)

## [1] 3879822 9

head(all\_trips)

## # A tibble: 6 x 9  
## ride\_id started\_at ended\_at rideable\_type start\_station\_id  
## <chr> <dttm> <dttm> <chr> <dbl>  
## 1 221785… 2019-04-01 00:02:22 2019-04-01 00:09:48 6251 81  
## 2 221785… 2019-04-01 00:03:02 2019-04-01 00:20:30 6226 317  
## 3 221785… 2019-04-01 00:11:07 2019-04-01 00:15:19 5649 283  
## 4 221785… 2019-04-01 00:13:01 2019-04-01 00:18:58 4151 26  
## 5 221785… 2019-04-01 00:19:26 2019-04-01 00:36:13 3270 202  
## 6 221785… 2019-04-01 00:19:39 2019-04-01 00:23:56 3123 420  
## # … with 4 more variables: start\_station\_name <chr>, end\_station\_id <dbl>,  
## # end\_station\_name <chr>, member\_casual <chr>

str(all\_trips)

## tibble [3,879,822 × 9] (S3: tbl\_df/tbl/data.frame)  
## $ ride\_id : chr [1:3879822] "22178529" "22178530" "22178531" "22178532" ...  
## $ started\_at : POSIXct[1:3879822], format: "2019-04-01 00:02:22" "2019-04-01 00:03:02" ...  
## $ ended\_at : POSIXct[1:3879822], format: "2019-04-01 00:09:48" "2019-04-01 00:20:30" ...  
## $ rideable\_type : chr [1:3879822] "6251" "6226" "5649" "4151" ...  
## $ start\_station\_id : num [1:3879822] 81 317 283 26 202 420 503 260 211 211 ...  
## $ start\_station\_name: chr [1:3879822] "Daley Center Plaza" "Wood St & Taylor St" "LaSalle St & Jackson Blvd" "McClurg Ct & Illinois St" ...  
## $ end\_station\_id : num [1:3879822] 56 59 174 133 129 426 500 499 211 211 ...  
## $ end\_station\_name : chr [1:3879822] "Desplaines St & Kinzie St" "Wabash Ave & Roosevelt Rd" "Canal St & Madison St" "Kingsbury St & Kinzie St" ...  
## $ member\_casual : chr [1:3879822] "Subscriber" "Subscriber" "Subscriber" "Subscriber" ...

summary(all\_trips)

## ride\_id started\_at ended\_at   
## Length:3879822 Min. :2019-04-01 00:02:22 Min. :2019-04-01 00:09:48   
## Class :character 1st Qu.:2019-06-23 07:49:09 1st Qu.:2019-06-23 08:20:27   
## Mode :character Median :2019-08-14 17:43:38 Median :2019-08-14 18:02:04   
## Mean :2019-08-26 00:49:59 Mean :2019-08-26 01:14:37   
## 3rd Qu.:2019-10-12 12:10:21 3rd Qu.:2019-10-12 12:36:16   
## Max. :2020-03-31 23:51:34 Max. :2020-05-19 20:10:34   
##   
## rideable\_type start\_station\_id start\_station\_name end\_station\_id   
## Length:3879822 Min. : 1.0 Length:3879822 Min. : 1.0   
## Class :character 1st Qu.: 77.0 Class :character 1st Qu.: 77.0   
## Mode :character Median :174.0 Mode :character Median :174.0   
## Mean :202.9 Mean :203.8   
## 3rd Qu.:291.0 3rd Qu.:291.0   
## Max. :675.0 Max. :675.0   
## NA's :1   
## end\_station\_name member\_casual   
## Length:3879822 Length:3879822   
## Class :character Class :character   
## Mode :character Mode :character   
##   
##   
##   
##

## 

## Examining The “members\_casual” Variable

Here the “$” is used to make a specific selection. In this case, we are taking the all\_trips DataFrame and selecting the members\_casual variable. Combining this with the table() command gives us the frequency of the different elements within this column.

table(all\_trips$member\_casual)

##   
## casual Customer member Subscriber   
## 48480 857474 378407 2595461

## 

## Recategorizing Users

The options for this category should be binary. We should only have “member” and “casual” as options for this variable. Instead, we have “Subscriber”, “Customer”, “member, and”casual“. The mutate() function is applied and the member\_casual column is selected. Recode() is used to switch all of the items listed as”Subscriber" to “member” and the ones listed as “Customer” are changed to “casual”.

all\_trips <- all\_trips %>%   
 mutate(member\_casual = recode(member\_casual  
 ,"Subscriber" = "member"  
 ,"Customer" = "casual"))

## 

## Confirming The Recategorization

We then rerun the table() function to confirm that we now only have two options for the variable member\_casual, as opposed to four.

table(all\_trips$member\_casual)

##   
## casual member   
## 905954 2973868

## 

## Adding Additional Columns

We are adding several new variables using an assignment operator. We can tell that we are adding a new column since we are using “$” to the left of the column names (which are names for new columns that do not currently exist within the all\_trips DataFrame). The as.date means that the new columns will be treated as dates. The “%m” shows this column will use the month part of the date field, “%d” is for day, etc.

all\_trips$date <- as.Date(all\_trips$started\_at)   
all\_trips$month <- format(as.Date(all\_trips$date), "%m")  
all\_trips$day <- format(as.Date(all\_trips$date), "%d")  
all\_trips$year <- format(as.Date(all\_trips$date), "%Y")  
all\_trips$day\_of\_week <- format(as.Date(all\_trips$date), "%A")

## 

## Calculating The Ride Length

As of now, we have no way of knowing how long each individual ride was. We can use the difftime() function to get the difference in time between the start of the ride “stared\_at” and the end “ended\_at”.

all\_trips$ride\_length <- difftime(all\_trips$ended\_at,all\_trips$started\_at)

## 

## Check The Structure of The all\_trips DataFrame

We are using the str() function to check the structure of the all\_trips DataFrame.

str(all\_trips)

## tibble [3,879,822 × 15] (S3: tbl\_df/tbl/data.frame)  
## $ ride\_id : chr [1:3879822] "22178529" "22178530" "22178531" "22178532" ...  
## $ started\_at : POSIXct[1:3879822], format: "2019-04-01 00:02:22" "2019-04-01 00:03:02" ...  
## $ ended\_at : POSIXct[1:3879822], format: "2019-04-01 00:09:48" "2019-04-01 00:20:30" ...  
## $ rideable\_type : chr [1:3879822] "6251" "6226" "5649" "4151" ...  
## $ start\_station\_id : num [1:3879822] 81 317 283 26 202 420 503 260 211 211 ...  
## $ start\_station\_name: chr [1:3879822] "Daley Center Plaza" "Wood St & Taylor St" "LaSalle St & Jackson Blvd" "McClurg Ct & Illinois St" ...  
## $ end\_station\_id : num [1:3879822] 56 59 174 133 129 426 500 499 211 211 ...  
## $ end\_station\_name : chr [1:3879822] "Desplaines St & Kinzie St" "Wabash Ave & Roosevelt Rd" "Canal St & Madison St" "Kingsbury St & Kinzie St" ...  
## $ member\_casual : chr [1:3879822] "member" "member" "member" "member" ...  
## $ date : Date[1:3879822], format: "2019-04-01" "2019-04-01" ...  
## $ month : chr [1:3879822] "04" "04" "04" "04" ...  
## $ day : chr [1:3879822] "01" "01" "01" "01" ...  
## $ year : chr [1:3879822] "2019" "2019" "2019" "2019" ...  
## $ day\_of\_week : chr [1:3879822] "Monday" "Monday" "Monday" "Monday" ...  
## $ ride\_length : 'difftime' num [1:3879822] 446 1048 252 357 ...  
## ..- attr(\*, "units")= chr "secs"

## 

## Formatting Ride Length Data Type

The as.numeric() command here ensures that when the assignment operator is applied, the ride\_length column in the DataFrame will be in numeric format.

is.factor(all\_trips$ride\_length)

## [1] FALSE

all\_trips$ride\_length <- as.numeric(as.character(all\_trips$ride\_length))  
is.numeric(all\_trips$ride\_length)

## [1] TRUE

## 

## Removing Invalid Data

We are using “!” to remove bad data by extracting items that are “==” exactly equal to the designation “HQ QR” or trips that were negative in duration. The final result is put in a new DataFrame called all\_trips\_V2.

all\_trips\_v2 <- all\_trips[!(all\_trips$start\_station\_name == "HQ QR" | all\_trips$ride\_length<0),]

## 

## Kicking Off Our Analysis With Some Basic Statistics

We’re specifying the DataFrame and column name within the parenthesis of a few functions so that we can generate some descriptive statistics.

mean(all\_trips\_v2$ride\_length)

## [1] 1479.139

median(all\_trips\_v2$ride\_length)

## [1] 712

max(all\_trips\_v2$ride\_length)

## [1] 9387024

min(all\_trips\_v2$ride\_length)

## [1] 1

If we want to use a quicker approach to do the same thing, we can just use the summary() function.

summary(all\_trips\_v2$ride\_length)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1 412 712 1479 1289 9387024

## 

## Members Vs. Casual Riders (Various Statistics)

We are using the function below to make a comparison between members and casual users.

aggregate(all\_trips\_v2$ride\_length ~ all\_trips\_v2$member\_casual, FUN = mean)

## all\_trips\_v2$member\_casual all\_trips\_v2$ride\_length  
## 1 casual 3552.7502  
## 2 member 850.0662

aggregate(all\_trips\_v2$ride\_length ~ all\_trips\_v2$member\_casual, FUN = median)

## all\_trips\_v2$member\_casual all\_trips\_v2$ride\_length  
## 1 casual 1546  
## 2 member 589

aggregate(all\_trips\_v2$ride\_length ~ all\_trips\_v2$member\_casual, FUN = max)

## all\_trips\_v2$member\_casual all\_trips\_v2$ride\_length  
## 1 casual 9387024  
## 2 member 9056634

aggregate(all\_trips\_v2$ride\_length ~ all\_trips\_v2$member\_casual, FUN = min)

## all\_trips\_v2$member\_casual all\_trips\_v2$ride\_length  
## 1 casual 2  
## 2 member 1

Here we are displaying average ride time as a function of the day of the week and members vs casual users.

aggregate(all\_trips\_v2$ride\_length ~ all\_trips\_v2$member\_casual + all\_trips\_v2$day\_of\_week, FUN = mean)

## all\_trips\_v2$member\_casual all\_trips\_v2$day\_of\_week all\_trips\_v2$ride\_length  
## 1 casual Friday 3773.8351  
## 2 member Friday 824.5305  
## 3 casual Monday 3372.2869  
## 4 member Monday 842.5726  
## 5 casual Saturday 3331.9138  
## 6 member Saturday 968.9337  
## 7 casual Sunday 3581.4054  
## 8 member Sunday 919.9746  
## 9 casual Thursday 3682.9847  
## 10 member Thursday 823.9278  
## 11 casual Tuesday 3596.3599  
## 12 member Tuesday 826.1427  
## 13 casual Wednesday 3718.6619  
## 14 member Wednesday 823.9996

## 

## Re-Ordering Day of The Week

This allows us to re-order days when they appear in output. Since the days didn’t come out in the order that we wanted them to, we can use the ordered() function, which helps us to rearrange them. The “levels=c” command lets us designate the actual order.

all\_trips\_v2$day\_of\_week <- ordered(all\_trips\_v2$day\_of\_week, levels=c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"))

## Ride Length Statistics

This shows the mean ride length as a function of member vs casual and day of the week.

aggregate(all\_trips\_v2$ride\_length ~ all\_trips\_v2$member\_casual + all\_trips\_v2$day\_of\_week, FUN = mean)

## all\_trips\_v2$member\_casual all\_trips\_v2$day\_of\_week all\_trips\_v2$ride\_length  
## 1 casual Sunday 3581.4054  
## 2 member Sunday 919.9746  
## 3 casual Monday 3372.2869  
## 4 member Monday 842.5726  
## 5 casual Tuesday 3596.3599  
## 6 member Tuesday 826.1427  
## 7 casual Wednesday 3718.6619  
## 8 member Wednesday 823.9996  
## 9 casual Thursday 3682.9847  
## 10 member Thursday 823.9278  
## 11 casual Friday 3773.8351  
## 12 member Friday 824.5305  
## 13 casual Saturday 3331.9138  
## 14 member Saturday 968.9337

We are breaking down ridership data by type and weekday.

all\_trips\_v2 %>%   
 mutate(weekday = wday(started\_at, label = TRUE)) %>%   
 group\_by(member\_casual, weekday) %>%   
 summarise(number\_of\_rides = n()   
 ,average\_duration = mean(ride\_length)) %>%   
 arrange(member\_casual, weekday)

## `summarise()` has grouped output by 'member\_casual'. You can override using the `.groups` argument.

## # A tibble: 14 x 4  
## # Groups: member\_casual [2]  
## member\_casual weekday number\_of\_rides average\_duration  
## <chr> <ord> <int> <dbl>  
## 1 casual Sun 181293 3581.  
## 2 casual Mon 103296 3372.  
## 3 casual Tue 90510 3596.  
## 4 casual Wed 92457 3719.  
## 5 casual Thu 102679 3683.  
## 6 casual Fri 122404 3774.  
## 7 casual Sat 209543 3332.  
## 8 member Sun 267965 920.  
## 9 member Mon 472196 843.  
## 10 member Tue 508445 826.  
## 11 member Wed 500329 824.  
## 12 member Thu 484177 824.  
## 13 member Fri 452790 825.  
## 14 member Sat 287958 969.

## 

## Visualization

This section displays setups for various visualizations designated by ggplot, which is a part of the tidyverse package. The “aes” is for aesthetic, which lets you choose your x & y axis and select some of your visual elements. The geom\_ part offers multiple options and it’s where you pick your chart type. In this case, geom\_col means you’re picking a column chart.

all\_trips\_v2 %>%   
 mutate(weekday = wday(started\_at, label = TRUE)) %>%   
 group\_by(member\_casual, weekday) %>%   
 summarise(number\_of\_rides = n()  
 ,average\_duration = mean(ride\_length)) %>%   
 arrange(member\_casual, weekday) %>%   
 ggplot(aes(x = weekday, y = number\_of\_rides, fill = member\_casual)) +  
 geom\_col(position = "dodge")

## `summarise()` has grouped output by 'member\_casual'. You can override using the `.groups` argument.



all\_trips\_v2 %>%   
 mutate(weekday = wday(started\_at, label = TRUE)) %>%   
 group\_by(member\_casual, weekday) %>%   
 summarise(number\_of\_rides = n()  
 ,average\_duration = mean(ride\_length)) %>%   
 arrange(member\_casual, weekday) %>%   
 ggplot(aes(x = weekday, y = average\_duration, fill = member\_casual)) +  
 geom\_col(position = "dodge")

## `summarise()` has grouped output by 'member\_casual'. You can override using the `.groups` argument.



## 

## Export

Here we are creating the DataFrame that we want to export, which we will call “counts”.

counts <- aggregate(all\_trips\_v2$ride\_length ~ all\_trips\_v2$member\_casual + all\_trips\_v2$day\_of\_week, FUN = mean)

Here we use write\_csv to make a csv file.

write.csv(counts, '/Users/cjmai/Documents/Everything Related to R/Google Data Analytics Bike Share Project 1\\DataAnalyticsVisualization.csv', row.names = FALSE)