Google Data Analytics: Capstone Project

```
library(tidyverse)  #helps wrangle data
```

```
## Warning: package 'tidyverse' was built under R version 4.0.5
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.1     v dplyr   1.0.6
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
```

```
## Warning: package 'ggplot2' was built under R version 4.0.5
```

```
## Warning: package 'tibble' was built under R version 4.0.5
```

```
## Warning: package 'tidyr' was built under R version 4.0.5
```

```
## Warning: package 'readr' was built under R version 4.0.5
```

```
## Warning: package 'purrr' was built under R version 4.0.5
```

```
## Warning: package 'dplyr' was built under R version 4.0.5
```

```
## Warning: package 'stringr' was built under R version 4.0.5
```

```
## Warning: package 'forcats' was built under R version 4.0.5
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)  #helps wrangle date attributes
```

```
## Warning: package 'lubridate' was built under R version 4.0.5
```

```
##
## Attaching package: 'lubridate'
```

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

```
library(ggplot2)  #helps visualize data
getwd() #displays your working directory
```

```
## [1] "D:/Google Data analytics/Data analytics capstone/Case-Study-1/Data"
```

Data

In 2016, Cyclistic launched a successful bike-share offering. Since then, the program has grown to a fleet of 5,824 bicycles that are geotracked and locked into a network of 692 stations across Chicago. The bikes can be unlocked from one station and returned to any other station in the system anytime.

Until now, Cyclistic's marketing strategy relied on building general awareness and appealing to broad consumer segments. One approach that helped make these things possible was the flexibility of its pricing plans: single-ride passes, full-day passes, and annual memberships. Customers who purchase single-ride or full-day passes are referred to as casual riders. Customers who purchase annual memberships are Cyclistic members.

Research Question

How do annual members and casual riders use Cyclistic bikes differently?

STEP 1: LOAD DATA

```
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()
## )
```

STEP 2: WRANGLE DATA AND COMBINE INTO A SINGLE FILE

```
# Compare column names of each of the files

colnames(q3_2019)
```

```
##  [1] "trip_id"           "start_time"        "end_time"
##  [4] "bikeid"            "tripduration"      "from_station_id"
##  [7] "from_station_name" "to_station_id"     "to_station_name"
## [10] "usertype"          "gender"            "birthyear"
```

```
colnames(q4_2019)
```

```
##  [1] "trip_id"           "start_time"        "end_time"
##  [4] "bikeid"            "tripduration"      "from_station_id"
##  [7] "from_station_name" "to_station_id"     "to_station_name"
## [10] "usertype"          "gender"            "birthyear"
```

```
colnames(q1_2020)
```

```
##  [1] "ride_id"            "rideable_type"      "started_at"
##  [4] "ended_at"           "start_station_name" "start_station_id"
##  [7] "end_station_name"   "end_station_id"     "start_lat"
## [10] "start_lng"          "end_lat"            "end_lng"
## [13] "member_casual"
```

```
# Rename columns  to make them consisent with q1_2020

(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>
```

```
# Inspect the dataframes and look for inconguencies
str(q1_2020)
```

```
## spec_tbl_df [426,887 x 13] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:426887] "EACB19130B0CDA4A" "8FED874C809DC021" "789F3C21E472C
A96" "C9A388DAC6ABF313" ...
##  $ rideable_type     : chr [1:426887] "docked_bike" "docked_bike" "docked_bike" "docked_bi
ke" ...
##  $ 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 x 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" "W
abash 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 x 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 00:0
1:16" ...
##  $ ended_at          : POSIXct[1:1640718], format: "2019-07-01 00:20:41" "2019-07-01 00:1
8:44" ...
##  $ 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 S
t" "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()
##   .. )
```

```
# Convert ride_id and rideable_type to character so that they can stack correctly
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))
```

```
# Stack individual quarter's data frames into one big data frame
all_trips <- bind_rows(q3_2019, q4_2019, q1_2020)
```

```
# Remove lat, long, birthyear, and gender fields as this data was dropped beginning in 2020
all_trips <- all_trips %>%
  select(-c(start_lat, start_lng, end_lat, end_lng, birthyear, gender))
```

STEP 3: CLEAN UP AND ADD DATA TO PREPARE FOR ANALYSIS

Inspect the new table that has been created

```
 #List of column names
colnames(all_trips)
```

```
##  [1] "ride_id"            "started_at"         "ended_at"
##  [4] "rideable_type"      "tripduration"       "start_station_id"
##  [7] "start_station_name" "end_station_id"     "end_station_name"
## [10] "member_casual"
```

```
 #How many rows are in data frame?
nrow(all_trips)
```

```
## [1] 2771659
```

```
#Dimensions of the data frame?
dim(all_trips)
```

```
## [1] 2771659      10
```

```
#See the first 6 rows of data frame.  Also tail(qs_raw)
head(all_trips)
```

```
## # A tibble: 6 x 10
##   ride_id  started_at          ended_at            rideable_type tripduration
##   <chr>    <dttm>              <dttm>              <chr>                <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
## # ... with 5 more variables: start_station_id <dbl>, start_station_name <chr>,
## #   end_station_id <dbl>, end_station_name <chr>, member_casual <chr>
```

```
 #See list of columns and data types (numeric, character, etc)
str(all_trips)
```

```
## tibble [2,771,659 x 10] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:2771659] "23479388" "23479389" "23479390" "23479391" ...
##  $ started_at        : POSIXct[1:2771659], format: "2019-07-01 00:00:27" "2019-07-01 00:0
1:16" ...
##  $ ended_at          : POSIXct[1:2771659], format: "2019-07-01 00:20:41" "2019-07-01 00:1
8:44" ...
##  $ rideable_type     : chr [1:2771659] "3591" "5353" "6180" "5540" ...
##  $ tripduration      : num [1:2771659] 1214 1048 1554 1503 1213 ...
##  $ start_station_id  : num [1:2771659] 117 381 313 313 168 300 168 313 43 43 ...
##  $ start_station_name: chr [1:2771659] "Wilton Ave & Belmont Ave" "Western Ave & Monroe S
t" "Lakeview Ave & Fullerton Pkwy" "Lakeview Ave & Fullerton Pkwy" ...
##  $ end_station_id    : num [1:2771659] 497 203 144 144 62 232 62 144 195 195 ...
##  $ end_station_name  : chr [1:2771659] "Kimball Ave & Belmont Ave" "Western Ave & 21st St"
"Larrabee St & Webster Ave" "Larrabee St & Webster Ave" ...
##  $ member_casual     : chr [1:2771659] "Subscriber" "Customer" "Customer" "Customer" ...
```

```
 #Statistical summary of data. Mainly for numerics
summary(all_trips)
```

```
##    ride_id            started_at                     ended_at
##  Length:2771659     Min.   :2019-07-01 00:00:27   Min.   :2019-07-01 00:07:31
##  Class :character   1st Qu.:2019-08-07 15:37:20   1st Qu.:2019-08-07 16:05:35
##  Mode  :character   Median :2019-09-14 18:29:24   Median :2019-09-14 19:02:20
##                     Mean   :2019-10-03 00:58:59   Mean   :2019-10-03 01:24:37
##                     3rd Qu.:2019-11-09 14:02:14   3rd Qu.:2019-11-09 14:26:07
##                     Max.   :2020-03-31 23:51:34   Max.   :2020-05-19 20:10:34
##
##  rideable_type       tripduration     start_station_id start_station_name
##  Length:2771659     Min.   :     61   Min.   :  2.0    Length:2771659
##  Class :character   1st Qu.:    423   1st Qu.: 77.0    Class :character
##  Mode  :character   Median :    732   Median :174.0    Mode  :character
##                     Mean   :   1576   Mean   :203.9
##                     3rd Qu.:   1322   3rd Qu.:291.0
##                     Max.   :9056633   Max.   :675.0
##                     NA's   :426887
##  end_station_id  end_station_name   member_casual
##  Min.   :  2.0   Length:2771659     Length:2771659
##  1st Qu.: 77.0   Class :character   Class :character
##  Median :175.0   Mode  :character   Mode  :character
##  Mean   :204.8
##  3rd Qu.:291.0
##  Max.   :675.0
##  NA's   :1
```

```
# Begin by seeing how many observations fall under each usertype
table(all_trips$member_casual)
```

```
##
##     casual   Customer     member Subscriber
##      48480     597888     378407    1746884
```

```
# Reassign to the desired values (we will go with the current 2020 labels)
all_trips <-  all_trips %>%
  mutate(member_casual = recode(member_casual
                                ,"Subscriber" = "member"
                                ,"Customer" = "casual"))
```

```
# Check to make sure the proper number of observations were reassigned
table(all_trips$member_casual)
```

```
##
##  casual  member
##  646368 2125291
```

Add columns that list the date, month, day, and year of each ride This will allow us to aggregate ride data for each month, day, or year before completing these operations we could only aggregate at the ride level.

```
all_trips$date <- as.Date(all_trips$started_at) #The default format is yyyy-mm-dd
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")
```

```
# Add a "ride_length" calculation to all_trips (in seconds)
all_trips$ride_length <- difftime(all_trips$ended_at,all_trips$started_at)
```

```
# Inspect the structure of the columns
str(all_trips)
```

```
## tibble [2,771,659 x 16] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:2771659] "23479388" "23479389" "23479390" "23479391" ...
##  $ started_at        : POSIXct[1:2771659], format: "2019-07-01 00:00:27" "2019-07-01 00:0
1:16" ...
##  $ ended_at          : POSIXct[1:2771659], format: "2019-07-01 00:20:41" "2019-07-01 00:1
8:44" ...
##  $ rideable_type     : chr [1:2771659] "3591" "5353" "6180" "5540" ...
##  $ tripduration      : num [1:2771659] 1214 1048 1554 1503 1213 ...
##  $ start_station_id  : num [1:2771659] 117 381 313 313 168 300 168 313 43 43 ...
##  $ start_station_name: chr [1:2771659] "Wilton Ave & Belmont Ave" "Western Ave & Monroe S
t" "Lakeview Ave & Fullerton Pkwy" "Lakeview Ave & Fullerton Pkwy" ...
##  $ end_station_id    : num [1:2771659] 497 203 144 144 62 232 62 144 195 195 ...
##  $ end_station_name  : chr [1:2771659] "Kimball Ave & Belmont Ave" "Western Ave & 21st St"
"Larrabee St & Webster Ave" "Larrabee St & Webster Ave" ...
##  $ member_casual     : chr [1:2771659] "member" "casual" "casual" "casual" ...
##  $ date              : Date[1:2771659], format: "2019-07-01" "2019-07-01" ...
##  $ month             : chr [1:2771659] "07" "07" "07" "07" ...
##  $ day               : chr [1:2771659] "01" "01" "01" "01" ...
##  $ year              : chr [1:2771659] "2019" "2019" "2019" "2019" ...
##  $ day_of_week       : chr [1:2771659] "Monday" "Monday" "Monday" "Monday" ...
##  $ ride_length       : 'difftime' num [1:2771659] 1214 1048 1554 1503 ...
##   ..- attr(*, "units")= chr "secs"
```

```
# Convert "ride_length" from Factor to numeric so we can run calculations on the data
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
```

Remove “bad” data.The dataframe includes a few hundred entries when bikes were taken out of docks and checked for quality by Divvy or ride_length was negative.We will create a new version of the dataframe (v2) since data is being removed

```
all_trips_v2 <- all_trips[!(all_trips$start_station_name == "HQ QR" | all_trips$ride_length<0
        ),]
```

STEP 4: CONDUCT DESCRIPTIVE ANALYSIS

Descriptive analysis on ride_length (all figures in seconds)

####straight average (total ride length / rides) ####midpoint number in the ascending array of ride lengths ####longest ride ####shortest ride

```
summary(all_trips_v2$ride_length)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1     406     701    1540    1265 9387024
```

```
# Compare 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                3812.0090
## 2                     member                 852.9544
```

```
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                      1505
## 2                     member                       583
```

```
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
```

```
# See the average ride time by each day for members vs casual users
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual + all_trips_v2$day_of_week, F
          UN = mean)
```

```
##    all_trips_v2$member_casual all_trips_v2$day_of_week all_trips_v2$ride_length
## 1                      casual                   Friday                4170.0055
## 2                      member                   Friday                 823.7175
## 3                      casual                   Monday                3632.0641
## 4                      member                   Monday                 843.2026
## 5                      casual                 Saturday                3460.4535
## 6                      member                 Saturday                 991.1996
## 7                      casual                   Sunday                3871.9759
## 8                      member                   Sunday                 910.9079
## 9                      casual                 Thursday                3928.9726
## 10                     member                 Thursday                 824.2099
## 11                     casual                  Tuesday                3872.3017
## 12                     member                  Tuesday                 837.5502
## 13                     casual                Wednesday                4007.4516
## 14                     member                Wednesday                 822.4819
```

```
# We see that the days of the week are out of order. Let's fix that.
all_trips_v2$day_of_week <- ordered(all_trips_v2$day_of_week, levels=c("Sunday", "Monday", "T
          uesday", "Wednesday", "Thursday", "Friday", "Saturday"))
```

```
# Now, let's run the average ride time by each day for members vs casual users
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual + all_trips_v2$day_of_week, F
          UN = mean)
```

```
##    all_trips_v2$member_casual all_trips_v2$day_of_week all_trips_v2$ride_length
## 1                      casual                  Sunday                3871.9759
## 2                      member                  Sunday                 910.9079
## 3                      casual                  Monday                3632.0641
## 4                      member                  Monday                 843.2026
## 5                      casual                 Tuesday                3872.3017
## 6                      member                 Tuesday                 837.5502
## 7                      casual               Wednesday                4007.4516
## 8                      member               Wednesday                 822.4819
## 9                      casual                Thursday                3928.9726
## 10                     member                Thursday                 824.2099
## 11                     casual                  Friday                4170.0055
## 12                     member                  Friday                 823.7175
## 13                     casual                Saturday                3460.4535
## 14                     member                Saturday                 991.1996
```

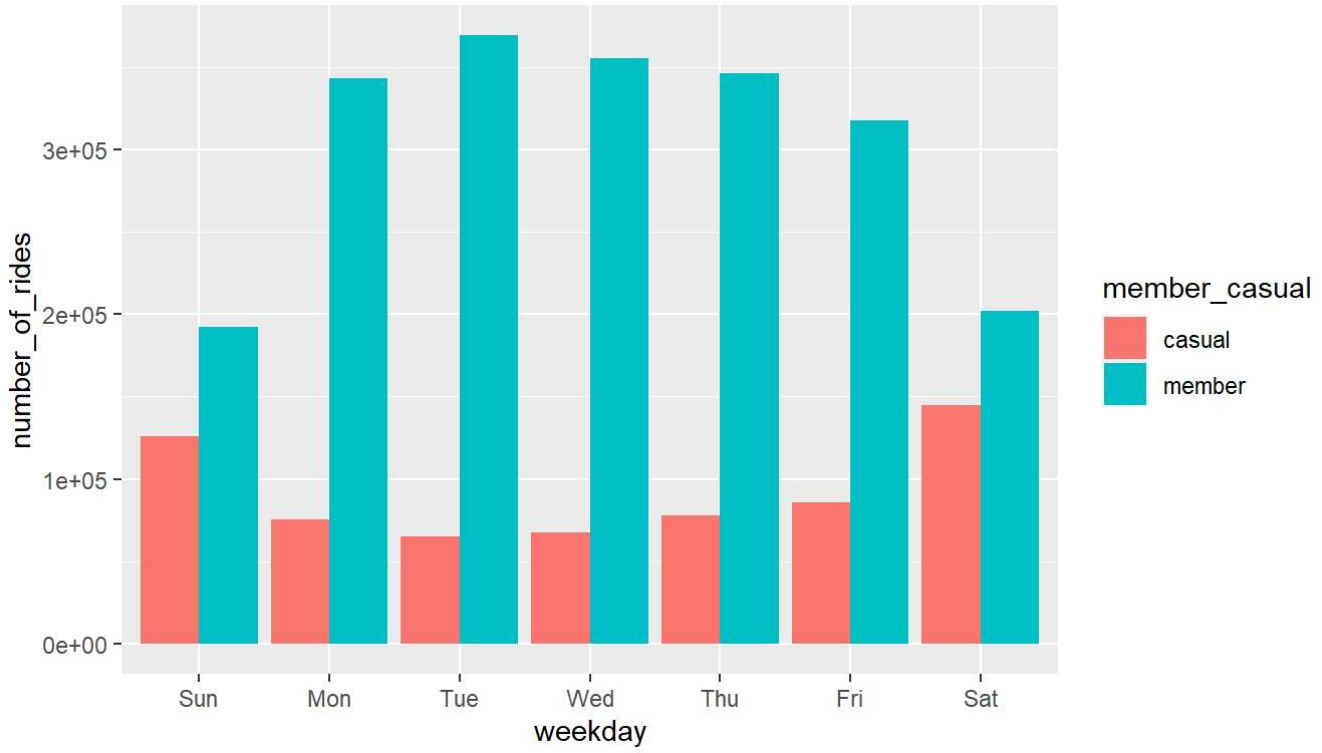
```
# analyze ridership data by type and weekday
all_trips_v2 %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, weekday) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(ride_length)) %>%
  arrange(member_casual, weekday)
```

```
## `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              125961            3872.
##  2 casual        Mon               75719            3632.
##  3 casual        Tue               64944            3872.
##  4 casual        Wed               67556            4007.
##  5 casual        Thu               77908            3929.
##  6 casual        Fri               85796            4170.
##  7 casual        Sat              144712            3460.
##  8 member        Sun              192315             911.
##  9 member        Mon              343330             843.
## 10 member        Tue              368924             838.
## 11 member        Wed              355026             822.
## 12 member        Thu              346228             824.
## 13 member        Fri              317386             824.
## 14 member        Sat              202074             991.
```

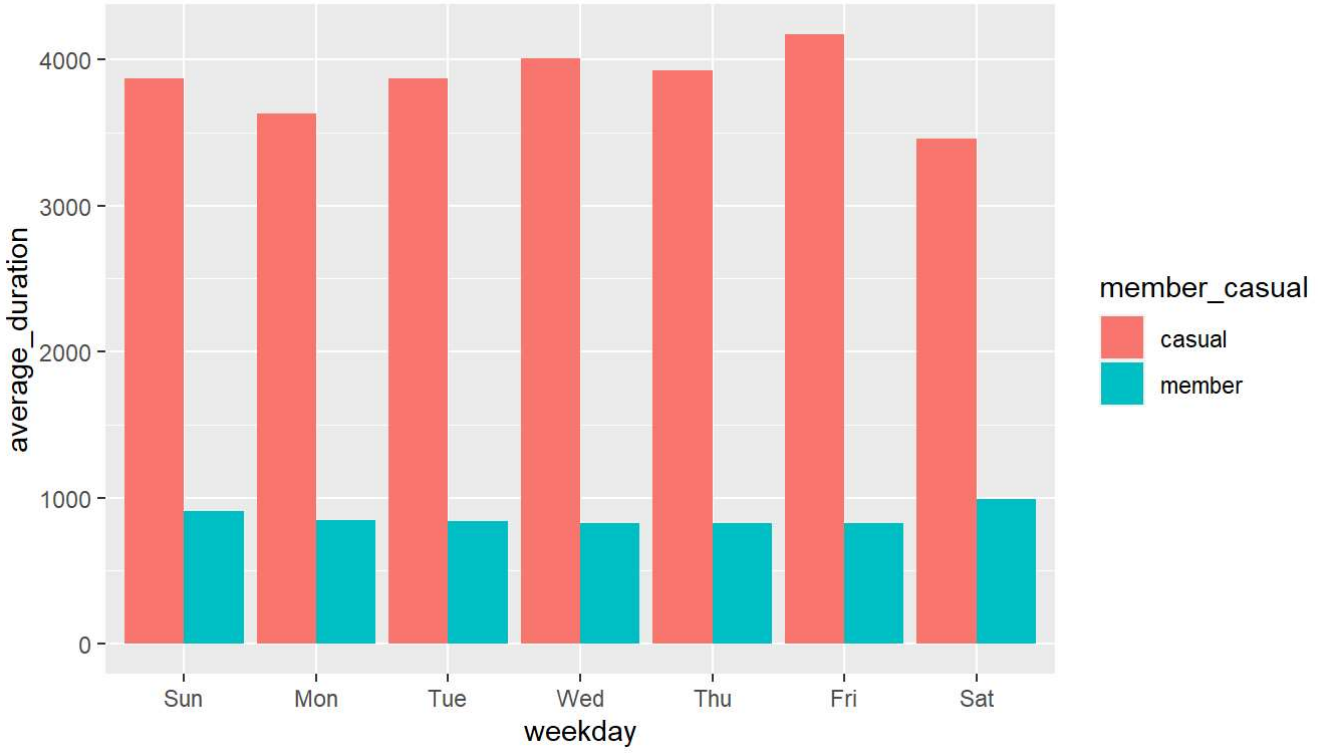
```
# Let's visualize the number of rides by rider type
all_trips_v2 %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, weekday) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(ride_length)) %>%
  arrange(member_casual, weekday)  %>%
  ggplot(aes(x = 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.
```




```
# Let's create a visualization for average duration
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.
```



STEP 5: EXPORT SUMMARY FILE FOR FURTHER ANALYSIS

Create a csv file that we will visualize in Excel, Tableau, or my presentation software

```
counts <- aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual + all_trips_v2$day_
        of_week, FUN = mean)
```

```
write.csv(counts, file = 'D:/Google Data analytics/Data analytics capstone/Case-Study-1.csv')
```