INTRODUCTION

Hello everyone, I just finished taking my Google Data Analytics Professional Certificate from Coursera, and now I would like to share what I have learned about data analytics throughout the entire course by showing my thoughts about a capstone project that they are providing and I choose a case study that is all about a cyclist-bike share company from Chicago.

To answer key business questions, I followed the six steps of the data analysis process taught in the course which is: **Ask, Prepare, Process, Analyze, Share and Act**.

BACKGROUND INFORMATION

You are a junior data analyst working in the marketing analyst team at Cyclistic, a bike-sharing company, Bikes can be unlocked from one station to another system anytime.

CHARACTERS AND TEAMS

Cyclistic: A bike-share program that features more than 5,800 bicycles and 600 docking stations. Cyclistic sets itself apart by also offering reclining bikes, hand tricycles, and cargo bikes. The majority of riders opt for traditional bikes; about 8% of riders use assistive options. Cyclistic users are more likely to ride for leisure, but about 30% use them to commute to work each day.
Lily Moreno: The director of marketing and your manager. Moreno is responsible for the development of campaigns and initiatives to promote the bike-share program.
Cyclistic marketing analytics team: A team of data analysts who are responsible for collecting, analyzing, and reporting data that helps guide Cyclistic marketing strategy
Cyclistic executive team: The notoriously detail-oriented executive team will decide whether to approve the recommended marketing program.

Cyclistic has flexible riding 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 buy annual memberships are Cyclistic members. Lily Moreno believes the company's future success depends on maximizing the number of annual memberships compared to casual riders. Moreno also believes that there is a very good chance to convert casual riders into members. She notes that casual riders are already aware of the Cyclistic program and have chosen Cyclistic for their mobility needs.

1. ASK
Identifying business task:
To analyze user behaviors on how annual members and casual riders use Cyclistic bikes differently to make recommendations on how to convert casual riders into annual members.

Consider stakeholders:
Lily Moreno and the Executive team

2. Prepare
The past 12 months of original bike-share dataset from **01/06/2021** to **30/22/2021** were extracted as [12 zipped .csv files]. The data is made available and licensed by Motivate International Inc.

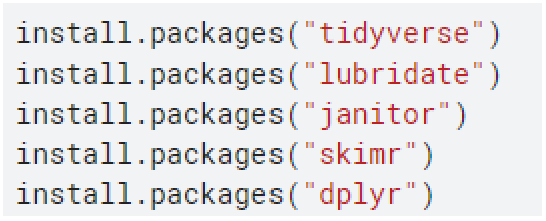
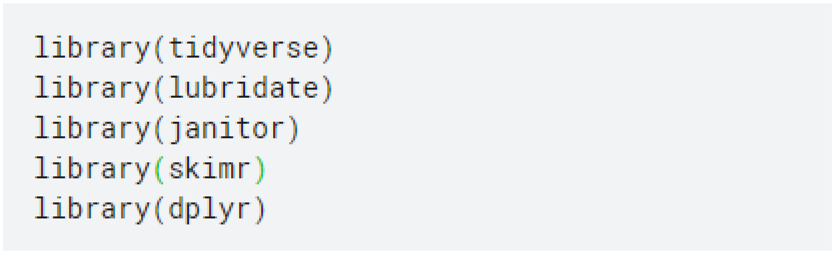
Data Organization and Description

File naming conventions: 2020_05-divvytripdata.csv
File type: converted to CSV file to xlsx to enable importation to RStudio
File Content: Each excel file contains 13 columns containing information related to ride id, ridership type, ride time and location etc. The number of rows varies between 49k to 531k from different excel files.

Data Limitation
Since the data-privacy issues prohibit us from using riders' personally identifiable information it means that we will not be able to connect pass purchases to credit card numbers to determine if casual riders live in the Cyclistic service area or if they have purchased multiple single passes.

Tools selected for data verification and cleaning:
1. Original files are backed up in a separate folder
2. Microsoft Excel
3. RStudio

REASON:
1. By scanning through Microsoft Excel will help me to familiarize myself on the general outline and basic information that can be found inside the dataset. And also i can perform a simple check on formatting, missing information, sorting, and filtering from the spreadsheet as well.
2. RStudio will be used to perform such tasks to generate and extract a new table for desired information which will be used for data visualization

3. Process
STEP 1. Setting up the R environment
Install necessary packages into the R console

```
install.packages("tidyverse")
install.packages("lubridate")
install.packages("janitor")
install.packages("skimr")
install.packages("dplyr")
```
Load the necessary packages after installing

```
library(tidyverse)
library(lubridate)
library(janitor)
library(skimr)
library(dplyr)
```
STEP 2: IMPORTING DATA
Uploaded 12 .csv files to directory. Each .csv file represents 1 month of Cyclistic data. Assigning new names & read .csv files

```
june_2021 <- read_csv("2021_06-divvy-tripdata.csv")
july_2021 <- read_csv("2021_07-divvy-tripdata.csv")
aug_2021 <- read_csv("2021_08-divvy-tripdata.csv")
sep_2021 <- read_csv("2021_09-divvy-tripdata.csv")
oct_2021 <- read_csv("2021_10-divvy-tripdata.csv")
nov_2021 <- read_csv("2021_11-divvy-tripdata.csv")
dec_2021 <- read_csv("2021_12-divvy-tripdata.csv")
jan_2022 <- read_csv("2022_01-divvy-tripdata.csv")
feb_2022 <- read_csv("2022_02-divvy-tripdata.csv")
mar_2022 <- read_csv("2022_03-divvy-tripdata.csv")
apr_2022 <- read_csv("2022_04-divvy-tripdata.csv")
may_2022 <- read_csv("2022_05-divvy-tripdata.csv")
```

STEP 3: CONSOLIDATE DATA

Here, we are making sure that all imported files have uniform column names prior to combining them into a singular file.

```
colnames(june_2021)
colnames(july_2021)
colnames(aug_2021)
colnames(sep_2021)
colnames(oct_2021)
colnames(nov_2021)
colnames(dec_2021)
colnames(jan_2022)
colnames(feb_2022)
colnames(mar_2022)
colnames(apr_2022)
colnames(may_2022)
```

Joining data frames

After ensuring all column names are uniform, we will be joining data frames into a singular file

```
all_trips <- bind_rows(june_2021, july_2021, aug_2021, sep_2021,
oct_2021, nov_2021, dec_2021, jan_2022,
feb_2022, mar_2022, apr_2022, may_2022)
```

Error: Can't combine `start_station_id` and `start_station_id` .

while joining them into a singular file hence here, I locate and manage to fix error mentioned above, the error code above prevented the join. Ran glimpse() on dataframes and identified where inconsistency was occurring

```
glimpse(june_2021)
glimpse(july_2021)
glimpse(aug_2021)
glimpse(sep_2021)
glimpse(oct_2021)
glimpse(nov_2021)
glimpse(dec_2021)
glimpse(jan_2022)
glimpse(feb_2022)
glimpse(mar_2022)
glimpse(apr_2022)
glimpse(may_2022)
```

The following dataframes were identified as housing the error:
nov_2021
oct_2021
sep_2021
aug_2021
jul_2021
jun_2021
may_2022
the same error was preemptively identified in 'end_station_id'

Therefore, both 'start_station_id' and 'end_station_id' were mutated to fix the error
so after determining the cause of errors we mutate the following:

```
nov_2021 <- mutate(nov_2021, start_station_id = as.character(start_station_id) ,
                   end_station_id = as.character(end_station_id))
oct_2021 <- mutate(oct_2021, start_station_id = as.character(start_station_id) ,
                   end_station_id = as.character(end_station_id))
sep_2021 <- mutate(sep_2021, start_station_id = as.character(start_station_id) ,
                   end_station_id = as.character(end_station_id))
aug_2021 <- mutate(aug_2021, start_station_id = as.character(start_station_id) ,
                   end_station_id = as.character(end_station_id))
jul_2021 <- mutate(jul_2021, start_station_id = as.character(start_station_id) ,
                   end_station_id = as.character(end_station_id))
jun_2021 <- mutate(jun_2021, start_station_id = as.character(start_station_id) ,
                   end_station_id = as.character(end_station_id))
may_2022 <- mutate(may_2022, start_station_id = as.character(start_station_id) ,
                   end_station_id = as.character(end_station_id))
```

after errors were confirmed to be fixed by the successful joining of all dataframes into a new aggregate dataframe named 'all_trips'

```
all_trips <- bind_rows(june_2021, july_2021, aug_2021, sep_2021,
oct_2021, nov_2021, dec_2021, jan_2022,
feb_2022, mar_2022, apr_2022, may_2022)
```

Removing Unnecessary Data (Latitude & Longitude)
Since we won't be able to use the latitude & longitude columns during our analysis, removing them is the best way to improve our analysis in the future.

These are the following columns that will be filtered:
‘start_lat’
‘start_lng’
‘end_lat’
‘end_lng’

```
all_trips <- all_trips %>%
select(-c(start_lat, start_lng, end_lat, end_lng))
```

STEP 4: CHECKPOINT
Since there are still a lot of things we need to process during cleaning and validating process of our dataset saving our new aggregate dataframe will help us to track our progress and also will be our checkpoint

```
save(all_trips, file = "all_trips.RData")
```

STEP 5: FORMATTING AND FORMULATING DATA

The first thing we want to do here is for us to separate the days, months, and year for us to have a better insight during our analysis

```
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")
```

Creating ride length observation as ‘ride_length’

Ride length will be a crucial observation for our analysis, here we use difftime to calculate the difference between ended_at and started_at under the ride length

```
all_trips$ride_length <- difftime(all_trips$ended_at,all_trips$started_at)
```

Assigning numeric data type to “ride_length” to enable calculations
Here I used factor() in R language for us to check if the object passed to the function is a Factor or not, then i also used numeric() to convert a character vector into a numeric vector.

```
is.factor(all_trips$ride_length)
all_trips$ride_length <- as.numeric(as.character(all_trips$ride_length))
is.numeric(all_trips$ride_length)
str(all_trips)
```

Formatting ride_length units from seconds to minutes, and to show only 1 decimal place, and removing units by setting them as numeric, be able to help us to ease our visual interpretation of data & improve the aesthetics of my presentation.

```
all_trips$ride_length <- format(as.numeric(difftime(all_trips$ended_at,all_trips$started_at, units = ("mins"))), digit s = 1, nsmall = 1)
```

Reassigning numeric data type to “ride_length” to enable calculations

```
is.factor(all_trips$ride_length)
all_trips$ride_length <- as.numeric(as.character(all_trips$ride_length))
is.numeric(trips_total$ride_length)
str(all_trips)
```

STEP 6: CLEANING DATA
After viewing newly created ride_length columns, the following outliers were identified when arranging data in ascending & descending order

Outliers:
Negative ride times
Trips not lasting more than 1 minute
Trips lasting over 1 day

Notes:
Negative ride times could be attributed to errors at a particular station or bike
Trips not lasting more than 1 minute can be discarded (ASSUMPTION)
Trips lasting more than 1 day can also be discarded (ASSUMPTION)
Start Station names with ‘HQ QR’ were bikes sent back for maintenance

Filter outliers & name to new dataframe
Filtering above the outliers and also renaming our new dataframe “all_trips_filtered”

```
all_trips_filtered <- all_trips[!(all_trips$start_station_name == "HQ QR" | all_trips$ride_length<1 | all_trips$ride_length>1440),]
```

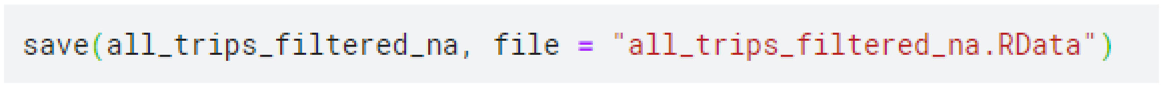
Filtering rows that contain NA values & name to new dataframe by using omit() to omit all unnecessary cases from data frame, matrix or vector.

```
all_trips_filtered_na = na.omit(all_trips_filtered)
```

Filtering rows that contain duplicate values by using distinct().

```
all_trips_filtered_na <- all_trips_filtered_na %>% distinct()
```

STEP 7: CHECKPOINT
Saving filtered dataframes, just like in step 4 this is used to saved and mark as our checkpoint



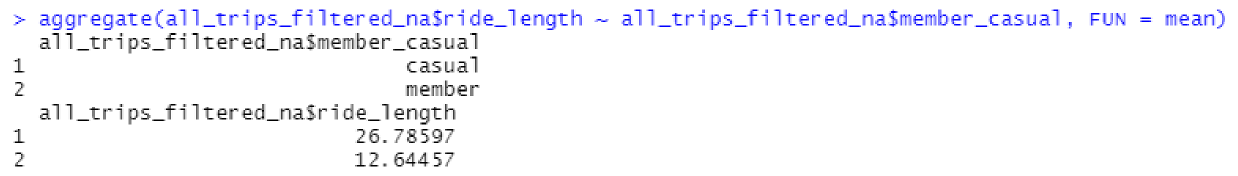
```
save(all_trips_filtered_na, file = "all_trips_filtered_na.RData")
```

STEP 8: ANALYZE

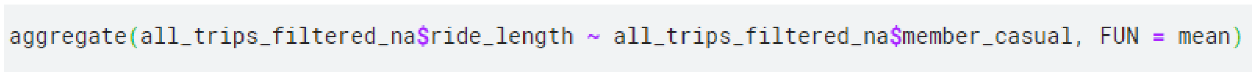
This is a very important step because here we will be going to determine the difference between casual vs. member in terms of average trip length, median trip length, average ride time by day and we will be comparing their number of rides & average ride length duration by weekday.

Average trip length

Results: Casual = 26.78597 minutes Member = 12.64457 minutes



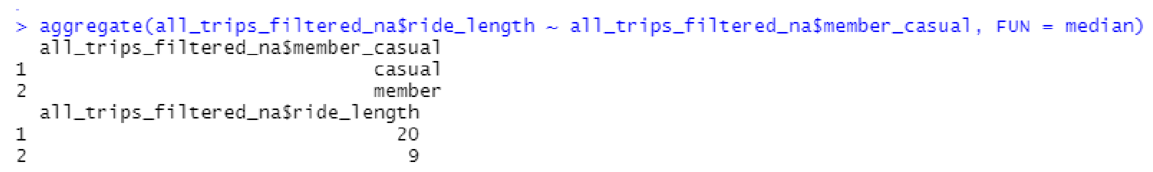
```
> aggregate(all_trips_filtered_na$ride_length ~ all_trips_filtered_na$member_casual, FUN = mean)
  all_trips_filtered_na$member_casual
1                              casual
2                              member
  all_trips_filtered_na$ride_length
1                          26.78597
2                          12.64457
```



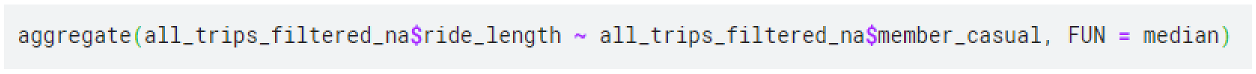
```
aggregate(all_trips_filtered_na$ride_length ~ all_trips_filtered_na$member_casual, FUN = mean)
```

Median Trip length

Results: Casual = 20 minutes Member = 9 minutes



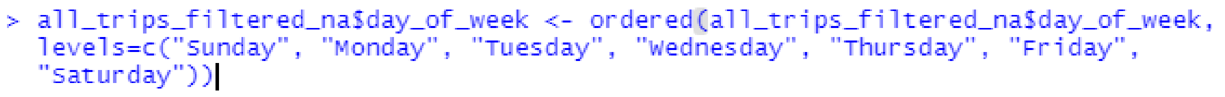
```
> aggregate(all_trips_filtered_na$ride_length ~ all_trips_filtered_na$member_casual, FUN = median)
  all_trips_filtered_na$member_casual
1                              casual
2                              member
  all_trips_filtered_na$ride_length
1                                20
2                                 9
```



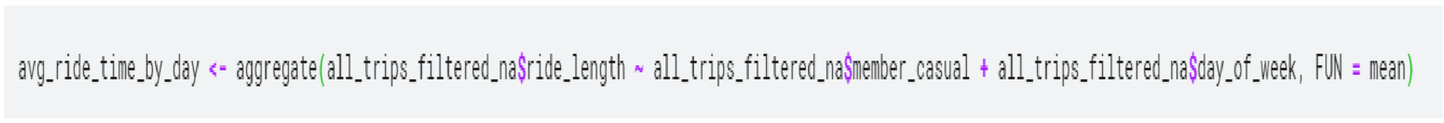
```
aggregate(all_trips_filtered_na$ride_length ~ all_trips_filtered_na$member_casual, FUN = median)
```

Average ride time by day

Run first line of code below to properly order the days of week in results section then use the second one is to get the average ride time by day



```
> all_trips_filtered_na$day_of_week <- ordered(all_trips_filtered_na$day_of_week,
  levels=c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday",
  "Saturday"))
```



```
avg_ride_time_by_day <- aggregate(all_trips_filtered_na$ride_length ~ all_trips_filtered_na$member_casual + all_trips_filtered_na$day_of_week, FUN = mean)
```

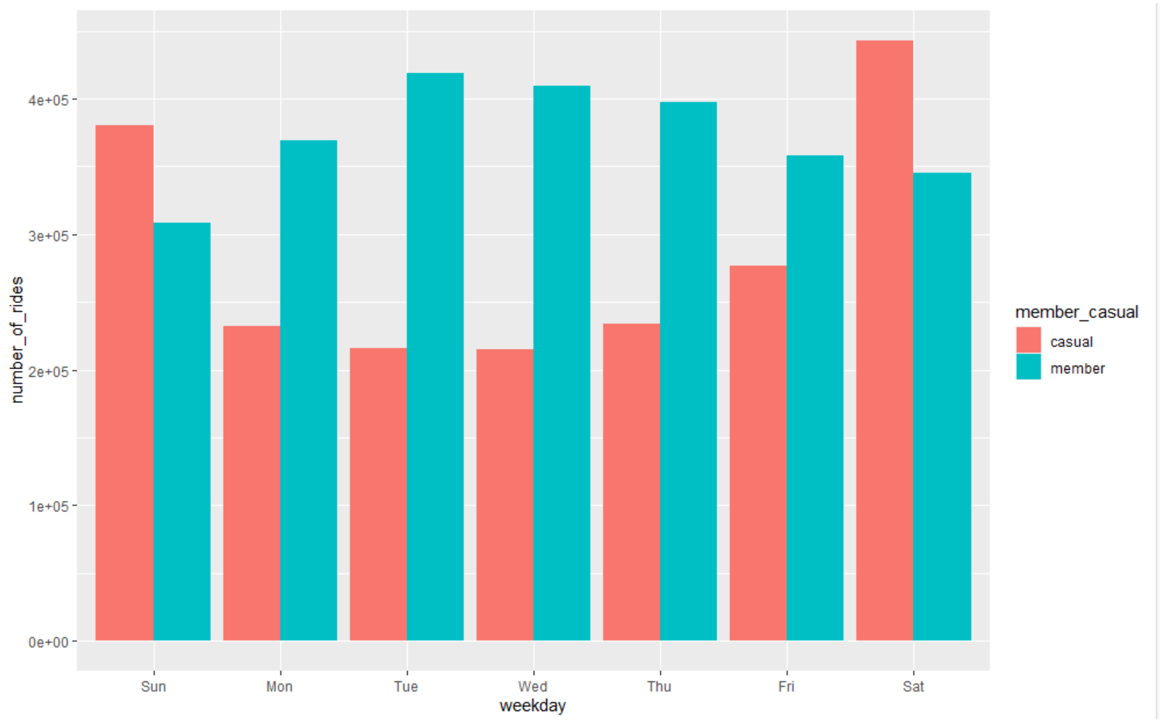
Comparing number of rides & average ride length duration by weekday

```
all_trips_filtered_na %>%
mutate(weekday = wday(started_at, label = TRUE)) %>% #creates weekday field using wday()
group_by(member_casual, weekday) %>%  #groups by usertype and weekday
summarise(number_of_rides = n(), average_duration = mean(ride_length)) %>% #calculates the number of rides and average duration
arrange(member_casual, weekday) %>% # calculates the average duration
```

STEP 9: VISUALIZE

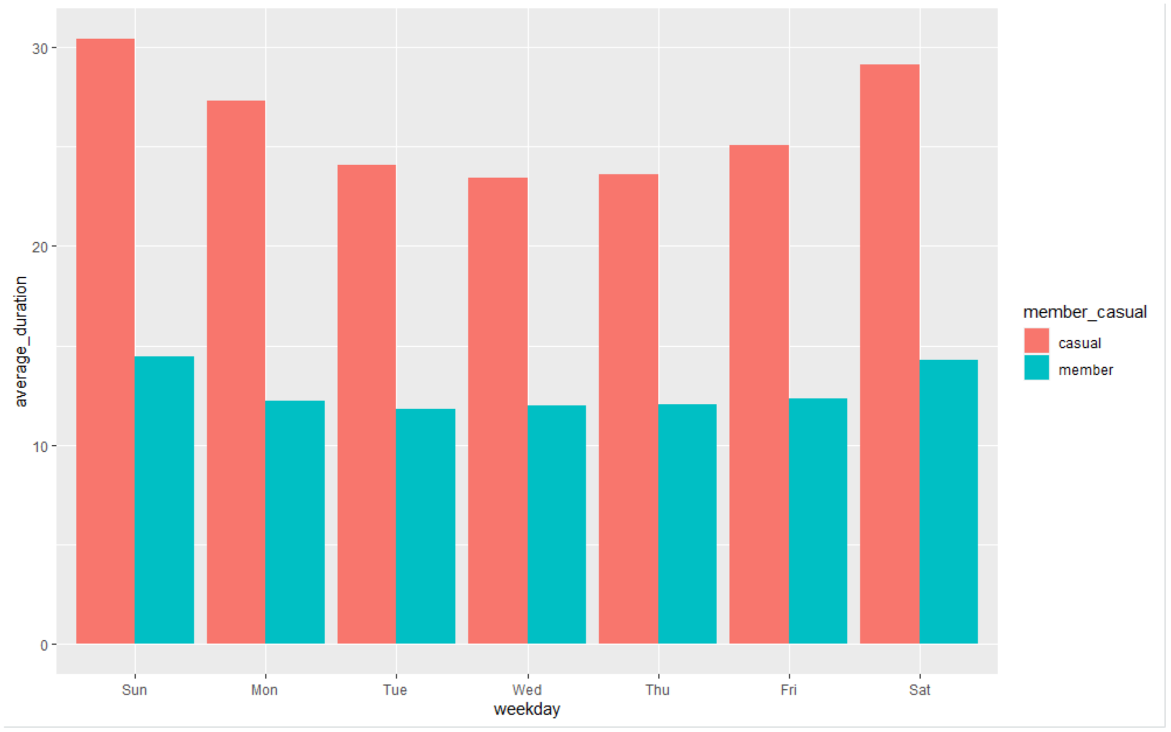
Number of rides by rider type

```
all_trips_filtered_na %>%
  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")
```



Average duration of ride length

```
all_trips_filtered_na %>%
    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")
```



5. ACT

Summary on the insight based from RStudio Data Visualization

1. Casual members are only active during weekends while during weekdays it drops the total number of casual rider type members, while members are mostly active weekdays based on the graph shown above regarding the number of rides based by rider type.
2. While on the second graph regarding the average duration based of ride length, casual members are more superior on average duration on usage of bike rentals while members have a low number of average duration during weekdays compared to casual that as shown in average duration of ride length chart.

In additional to sharing the insights gathered to Lily Moreno and the executive stakeholder. I would like to propose a few recommendations based on data evidence:

1. As casual rider usage peaks on weekend, the marketing campaign can include weekend only membership subscription at lower price to attract casual riders to convert to members
2. Modification of membership subscription such as ride length based charges must charge lesser as ride length increases. This provides more incentive for the member rides to cycle longer distances. With these modification they can be encourage casual riders to convert to members to enjoy ride length discounts.

REFERENCES:
1**.** Song Jiahe. (2021). *Google Data Analytics Professional Certificate Capstone Project.* https://jsong91.medium.com/google-data-analytics-professional-certificate-capstone-project-498e4c1e4d52
2. Wen Hao Tan. (2021). *Google Data Analytics Course Capstone Project*. https://www.linkedin.com/pulse/google-data-analytics-course-capstone-project-wen-hao-tan/
3. Joseph Powers (2021). *Cyclistic Capstone Project.* https://rpubs.com/JPowers/788296