Google Capstone Project - Cyclistic Case Study

R Welsh

2024-09-25

The Background

Scenario

You are a junior data analyst working on the marketing analyst team at Cyclistic, a bike-share company in Chicago. The director of marketing believes the company's future success depends on maximizing the number of annual memberships. Therefore,your team wants to understand how casual riders and annual members use Cyclistic bikes differently. From these insights, your team will design a new marketing strategy to convert casual riders into annual members. But first, Cyclistic executives must approve your recommendations, so they must be backed up with compeling data insights and professional data visualizations.

The Stakeholders 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, making bike-share more inclusive to people with disabilities and riders who can't use a standard two-wheeled bike. The majority of riders opt for traditional bikes; about 8% of riders use the assistive options. Cyclistic users are more likely to ride for leisure, but about 30% use the bikes 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.These may include email, social media, and other channels.

Cyclistic marketing analytics team: A team of data analysts who are responsible for collecting, analyzing, and reporting data that helps guide Cyclistic marketing strategy. You joined this team six months ago and have been busy learning about Cyclistic's mission and business goals— aswell as how you, as a junior data analyst, can help Cyclistic achieve them.

Cyclistic executive team: The notoriously detail-oriented executive team will decide whether to approve the recommended marketing program.

Company Overview

In 2016, Cyclistic launched a successful bike-share ordering service. 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 exibility 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. Cyclistic's analysts have concluded that annual members are much more profitable than casual riders. Although the pricing exibility helps Cyclistic attract more customers, Moreno believes that maximizing the

number of annual members will be key to future growth. Rather than creating a marketing campaign that targets new customers, Moreno believes there is a solid opportunity 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.

Moreno has set a clear goal: Design marketing strategies aimed at converting casual riders into annual members. In order to do that, however, the team needs to better understand how annual members and casual riders differ, why casual riders would buy a membership, and how digital media could affect their marketing tactics. Moreno and her team are interested in analyzing the Cyclistic historical bike trip data to identify trends.

My Role

Moreno has assigned me to see how annual members and casual members differ in their use of the service.

How will I do this

First thing I will need to do is get a dataset that will be used to help my derive analysis to therfore make a report on my findings.

Gathering the datasets required for my analyis

As Cyclistic is a "mock" company there is no first party data so I gathered the datasets required for this project from this link https://divvy-tripdata.s3.amazonaws.com/index.html. This data has been made available from Motivate International Inc Under the license https://divvybikes.com/data-license-agreement The datasets I have selected are (2019 Q1 - 2020 Q1) as these two datasets cover the same months therefore I am able to make a better correlation based on the average use i.e certain months later in the quarter may have a higher average due to better weather conditions. This will also give a better indication to the marketing team if they were to advertise in a specific period Q1 what type of indicators they will need to target.

Preparing and Cleaning the data

After downloading and unzipping the datasets required I opened each spreadsheet in Excel and checked for duplicated values among the data, No duplicated values were found therefore the remove duplicates function didn't need to be used.

I made sure the datasets had the same headers and values this meant changing the header "usertype" from the 2019 Q1 sheet to "member_causal" and also the values "subscriber" to "Annual" and "customer" to "Casual". I then deleted columns that weren't across both sheets of data and that I would be unable to validate so the Gender and Birthday columns from 2019 Q1 were removed.

Start of the analysis

I then went through both of the datasets and decided to find the mean, the mode, the max and the min of the rides.

2020 Q1 issue During this process I found 210 records that had either 0 min or negative minute rides, this could be due to the tracking on these rides however there is a correlation that all these rides started at station 675. As I have no viable way to correct this data as this is a 3rd party data source I decided to remove these values.

Next Steps

I then added a weekday function in order to view the day the rides were made as this could be a way to differentiate how the types of user use the service. I then added a mode function to see what day was the most common.

Individual Sheet Analysis

I then made pivot tables and visualisations for each Q1 dataset to find and showcase the difference between Annual and Casual users. The 3 metrics I wanted to view was average ride length between the two members, the average ride length by day and the amount of use per member per day.

The spreadsheets are able to be viewed on my portfolio

Using R to Merge and Produce further analysis First thing to do when opening R is to open the packages I will need to provide analysis on the data

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(readxl)
library(janitor)
```

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

```
library(skimr)
library(flexdashboard)
```

```
library(tinytex)
```

To show the analysis I had to import the data I first had to change my working directory in R to where the Spreadsheets were held

```
setwd("C:/Documents/Google Capstone Project/Cyclistic Case Study")
```

Then I imported my clean datasets

```
Divvy_Trips_2019_Q1 <- read_excel("Divvy_Trips_2019_Q1.xlsx", sheet = "Divvy_Trips_2019_Q1")
```

```
Divvy_Trips_2020_Q1 <- read_excel("Divvy_Trips_2020_Q1.xlsx", sheet = "Divvy_Trips_2020_Q1")
```

I did have Issues with the code took a while to eventually work importing these datasets, My current thinking process is that this may be as I had changed my default directory. another way to import the datasets manually is by

1. Using the more file commands dropdown
2. Go to Working Directory
3. Click on the Two files and import

Next Steps

View the 2 datasets to see if they've imported correctly

```
View(Divvy_Trips_2019_Q1)
```

```
View(Divvy_Trips_2020_Q1)
```

check the column names of the two datasets

```
colnames(Divvy_Trips_2019_Q1)
```

```
##  [1] "trip_id"           "start_time"        "end_time"
##  [4] "bikeid"            "tripduration"      "from_station_id"
##  [7] "from_station_name" "to_station_id"     "to_station_name"
## [10] "member_casual"     "ride_length"       "Day of week"
## [13] "Mean_ride_length"  "Mode_ride_length"  "Max_ride_length"
## [16] "Min_ride_length"   "Mode Weekday"
```

```
colnames(Divvy_Trips_2020_Q1)
```

```
##  [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"      "ride_length"        "Day of week"
## [16] "Mean_ride_length"   "Mode_ride_length"   "Max_ride_length"
## [19] "Min_ride_length"    "Mode_Weekday"
```

Then the head of both documents

```
head(Divvy_Trips_2019_Q1)
```

```
## # A tibble: 6 x 17
##    trip_id start_time          end_time            bikeid tripduration
##      <dbl> <dttm>              <dttm>               <dbl>        <dbl>
## 1 21742443 2019-01-01 00:04:37 2019-01-01 00:11:07   2167          390
```

```
## 2 21742444 2019-01-01 00:08:13 2019-01-01 00:15:34   4386          441
## 3 21742445 2019-01-01 00:13:23 2019-01-01 00:27:12   1524          829
## 4 21742446 2019-01-01 00:13:45 2019-01-01 00:43:28    252         1783
## 5 21742447 2019-01-01 00:14:52 2019-01-01 00:20:56   1170          364
## 6 21742448 2019-01-01 00:15:33 2019-01-01 00:19:09   2437          216
## # i 12 more variables: from_station_id <dbl>, from_station_name <chr>,
## #   to_station_id <dbl>, to_station_name <chr>, member_casual <chr>,
## #   ride_length <dttm>, `Day of week` <dbl>, Mean_ride_length <dttm>,
## #   Mode_ride_length <dttm>, Max_ride_length <dttm>, Min_ride_length <dttm>,
## #   `Mode Weekday` <dbl>
```

```
head(Divvy_Trips_2020_Q1)
```

```
## # A tibble: 6 x 20
##   ride_id          rideable_type started_at          ended_at
##   <chr>            <chr>         <dttm>              <dttm>
## 1 EACB19130B0CDA4A docked_bike   2020-01-21 20:06:59 2020-01-21 20:14:30
## 2 8FED874C809DC021 docked_bike   2020-01-30 14:22:39 2020-01-30 14:26:22
## 3 789F3C21E472CA96 docked_bike   2020-01-09 19:29:26 2020-01-09 19:32:17
## 4 C9A388DAC6ABF313 docked_bike   2020-01-06 16:17:07 2020-01-06 16:25:56
## 5 943BC3CBECCFD662 docked_bike   2020-01-30 08:37:16 2020-01-30 08:42:48
## 6 6D9C8A6938165C11 docked_bike   2020-01-10 12:33:05 2020-01-10 12:37:54
## # i 16 more variables: start_station_name <chr>, start_station_id <dbl>,
## #   end_station_name <chr>, end_station_id <dbl>, start_lat <dbl>,
## #   start_lng <dbl>, end_lat <dbl>, end_lng <dbl>, member_casual <chr>,
## #   ride_length <dttm>, `Day of week` <dbl>, Mean_ride_length <dttm>,
## #   Mode_ride_length <dttm>, Max_ride_length <dttm>, Min_ride_length <dttm>,
## #   Mode_Weekday <dbl>
```

As you can see I still need to amend and delete some column names in order to make both sheets workable when I merge them together.

Sorting the Datasets

First thing was renaming the columns in the 2019 Q1 Dataset so that they matched with the 2020 dataset

```
(Divvy_Trips_2019_Q1 <- rename(Divvy_Trips_2019_Q1
                   ,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
                   ))
```

```
## # A tibble: 365,069 x 17
##     ride_id started_at          ended_at            rideable_type tripduration
##       <dbl> <dttm>              <dttm>                      <dbl>        <dbl>
##  1 21742443 2019-01-01 00:04:37 2019-01-01 00:11:07          2167          390
##  2 21742444 2019-01-01 00:08:13 2019-01-01 00:15:34          4386          441
```

```
##  3 21742445 2019-01-01 00:13:23 2019-01-01 00:27:12          1524          829
##  4 21742446 2019-01-01 00:13:45 2019-01-01 00:43:28           252         1783
##  5 21742447 2019-01-01 00:14:52 2019-01-01 00:20:56          1170          364
##  6 21742448 2019-01-01 00:15:33 2019-01-01 00:19:09          2437          216
##  7 21742449 2019-01-01 00:16:06 2019-01-01 00:19:03          2708          177
##  8 21742450 2019-01-01 00:18:41 2019-01-01 00:20:21          2796          100
##  9 21742451 2019-01-01 00:18:43 2019-01-01 00:47:30          6205         1727
## 10 21742452 2019-01-01 00:19:18 2019-01-01 00:24:54          3939          336
## # i 365,059 more rows
## # i 12 more variables: start_station_id <dbl>, start_station_name <chr>,
## #   end_station_id <dbl>, end_station_name <chr>, member_casual <chr>,
## #   ride_length <dttm>, `Day of week` <dbl>, Mean_ride_length <dttm>,
## #   Mode_ride_length <dttm>, Max_ride_length <dttm>, Min_ride_length <dttm>,
## #   `Mode Weekday` <dbl>
```

As you can see this is now much more easier when I will be further analysing the data.

Next I will be converting the data from the 2019 Q1 dataset so that it matches the same datatype so it will stack correctly when merged

```
Divvy_Trips_2019_Q1 <-  mutate(Divvy_Trips_2019_Q1, ride_id = as.character(ride_id)
                   ,rideable_type = as.character(rideable_type))
```

Now it is time to merge the two datasets together into a new dataframe

```
all_trips <- bind_rows(Divvy_Trips_2019_Q1, Divvy_Trips_2020_Q1)
```

This has now created a new dataset called all_trips which I will view

```
View(all_trips)
```

check the columns are correct

```
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"      "ride_length"        "Day of week"
## [13] "Mean_ride_length"   "Mode_ride_length"   "Max_ride_length"
## [16] "Min_ride_length"    "Mode Weekday"       "start_lat"
## [19] "start_lng"          "end_lat"            "end_lng"
## [22] "Mode_Weekday"
```

And check the rows and columns have stacked correctly

```
head(all_trips)
```

```
## # A tibble: 6 x 22
##   ride_id  started_at          ended_at            rideable_type tripduration
##   <chr>    <dttm>              <dttm>              <chr>                 <dbl>
```

```
## 1 21742443 2019-01-01 00:04:37 2019-01-01 00:11:07 2167                    390
## 2 21742444 2019-01-01 00:08:13 2019-01-01 00:15:34 4386                    441
## 3 21742445 2019-01-01 00:13:23 2019-01-01 00:27:12 1524                    829
## 4 21742446 2019-01-01 00:13:45 2019-01-01 00:43:28 252                    1783
## 5 21742447 2019-01-01 00:14:52 2019-01-01 00:20:56 1170                    364
## 6 21742448 2019-01-01 00:15:33 2019-01-01 00:19:09 2437                    216
## # i 17 more variables: start_station_id <dbl>, start_station_name <chr>,
## #   end_station_id <dbl>, end_station_name <chr>, member_casual <chr>,
## #   ride_length <dttm>, `Day of week` <dbl>, Mean_ride_length <dttm>,
## #   Mode_ride_length <dttm>, Max_ride_length <dttm>, Min_ride_length <dttm>,
## #   `Mode Weekday` <dbl>, start_lat <dbl>, start_lng <dbl>, end_lat <dbl>,
## #   end_lng <dbl>, Mode_Weekday <dbl>
```

As you can see some columns are only reference in the Q1 2019 dataset therefore I will need to clean this new dataframe before beginning analysis.

```
all_trips <- all_trips %>%
  select(-c(start_lat, start_lng, end_lat, end_lng, `Day of week`, Max_ride_length, Mode_ride_length, r
```

Removing Columns I will not need

Time to Inspect my New Dataframe

Now it's time to inspect my new Dataframe.

```
View(all_trips)
```

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

```
## [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)  #How many rows are in data frame?
```

```
## [1] 788189
```

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

```
## [1] 788189      9
```

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

```
## # A tibble: 6 x 9
##   ride_id started_at          ended_at            rideable_type start_station_id
##   <chr>   <dttm>              <dttm>              <chr>                     <dbl>
```

```
## 1 217424~ 2019-01-01 00:04:37 2019-01-01 00:11:07 2167                      199
## 2 217424~ 2019-01-01 00:08:13 2019-01-01 00:15:34 4386                       44
## 3 217424~ 2019-01-01 00:13:23 2019-01-01 00:27:12 1524                       15
## 4 217424~ 2019-01-01 00:13:45 2019-01-01 00:43:28 252                       123
## 5 217424~ 2019-01-01 00:14:52 2019-01-01 00:20:56 1170                      173
## 6 217424~ 2019-01-01 00:15:33 2019-01-01 00:19:09 2437                       98
## # i 4 more variables: start_station_name <chr>, end_station_id <dbl>,
## #   end_station_name <chr>, member_casual <chr>
```

```
tail(all_trips)
```

```
## # A tibble: 6 x 9
##   ride_id started_at          ended_at            rideable_type start_station_id
##   <chr>   <dttm>              <dttm>              <chr>                     <dbl>
## 1 4361A2~ 2020-03-25 17:10:10 2020-03-25 17:47:34 docked_bike                 313
## 2 005D3B~ 2020-03-12 07:32:25 2020-03-12 07:38:44 docked_bike                  91
## 3 82B10F~ 2020-03-07 15:25:55 2020-03-07 16:14:03 docked_bike                 161
## 4 AA0D5A~ 2020-03-01 13:12:38 2020-03-01 13:38:29 docked_bike                 141
## 5 329636~ 2020-03-07 18:02:45 2020-03-07 18:13:18 docked_bike                 672
## 6 064EC7~ 2020-03-08 13:03:57 2020-03-08 13:32:27 docked_bike                 110
## # i 4 more variables: start_station_name <chr>, end_station_id <dbl>,
## #   end_station_name <chr>, member_casual <chr>
```

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

```
## tibble [788,189 x 9] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:788189] "21742443" "21742444" "21742445" "21742446" ...
##  $ started_at        : POSIXct[1:788189], format: "2019-01-01 00:04:37" "2019-01-01 00:08:13" ...
##  $ ended_at          : POSIXct[1:788189], format: "2019-01-01 00:11:07" "2019-01-01 00:15:34" ...
##  $ rideable_type     : chr [1:788189] "2167" "4386" "1524" "252" ...
##  $ start_station_id  : num [1:788189] 199 44 15 123 173 98 98 211 150 268 ...
##  $ start_station_name: chr [1:788189] "Wabash Ave & Grand Ave" "State St & Randolph St" "Racine Ave &
##  $ end_station_id    : num [1:788189] 84 624 644 176 35 49 49 142 148 141 ...
##  $ end_station_name  : chr [1:788189] "Milwaukee Ave & Grand Ave" "Dearborn St & Van Buren St (*)" "U
##  $ member_casual     : chr [1:788189] "Annual" "Annual" "Annual" "Annual" ...
```

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

```
##    ride_id            started_at
##  Length:788189      Min.   :2019-01-01 00:04:37.00
##  Class :character   1st Qu.:2019-02-28 13:39:58.00
##  Mode  :character   Median :2020-01-07 07:59:53.00
##                     Mean   :2019-08-31 14:14:43.81
##                     3rd Qu.:2020-02-19 12:38:46.00
##                     Max.   :2020-03-31 23:51:34.00
##     ended_at                      rideable_type      start_station_id
##  Min.   :2019-01-01 00:11:07.00   Length:788189      Min.   :  2.0
##  1st Qu.:2019-02-28 13:51:45.00   Class :character   1st Qu.: 77.0
##  Median :2020-01-07 08:10:57.00   Mode  :character   Median :174.0
##  Mean   :2019-08-31 14:34:33.26                      Mean   :202.2
##  3rd Qu.:2020-02-19 12:57:45.00                      3rd Qu.:289.0
##  Max.   :2020-05-19 20:10:34.00                      Max.   :673.0
```

```
##  start_station_name end_station_id  end_station_name   member_casual
##  Length:788189      Min.   :  2.0   Length:788189      Length:788189
##  Class :character   1st Qu.: 77.0   Class :character   Class :character
##  Mode  :character   Median :173.0   Mode  :character   Mode  :character
##                     Mean   :202.1
##                     3rd Qu.:289.0
##                     Max.   :675.0
```

Further Actions

I need to reintroduce the columns I had for the individual spreadsheets i.e Weekday, Ride Length, as well as create new columns i.e Day, Month, Year so that I am able to further aggregate the data.

Adding the Date,Month,Year,Day to each ride

The reason I want to add this to the Dataframe is in order to be able to gain more opportunities to aggragate the data.

```
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") # Month
```

```
all_trips$day <- format(as.Date(all_trips$date), "%d") # Day
```

```
all_trips$year <- format(as.Date(all_trips$date), "%Y") # Year
```

```
all_trips$day_of_week <- format(as.Date(all_trips$date), "%A") # Day of Week
```

Now my Dataframe looks like this,

```
head(all_trips)
```

```
## # A tibble: 6 x 14
##   ride_id started_at          ended_at            rideable_type start_station_id
##   <chr>   <dttm>              <dttm>              <chr>                     <dbl>
## 1 217424~ 2019-01-01 00:04:37 2019-01-01 00:11:07 2167                        199
## 2 217424~ 2019-01-01 00:08:13 2019-01-01 00:15:34 4386                         44
## 3 217424~ 2019-01-01 00:13:23 2019-01-01 00:27:12 1524                         15
## 4 217424~ 2019-01-01 00:13:45 2019-01-01 00:43:28 252                         123
## 5 217424~ 2019-01-01 00:14:52 2019-01-01 00:20:56 1170                        173
## 6 217424~ 2019-01-01 00:15:33 2019-01-01 00:19:09 2437                         98
## # i 9 more variables: start_station_name <chr>, end_station_id <dbl>,
## #   end_station_name <chr>, member_casual <chr>, date <date>, month <chr>,
## #   day <chr>, year <chr>, day_of_week <chr>
```

Adding the Ride Length (in Seconds)

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

Then I will inspect the structure of the column

```
str(all_trips)
```

```
## tibble [788,189 x 15] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:788189] "21742443" "21742444" "21742445" "21742446" ...
##  $ started_at        : POSIXct[1:788189], format: "2019-01-01 00:04:37" "2019-01-01 00:08:13" ...
##  $ ended_at          : POSIXct[1:788189], format: "2019-01-01 00:11:07" "2019-01-01 00:15:34" ...
##  $ rideable_type     : chr [1:788189] "2167" "4386" "1524" "252" ...
##  $ start_station_id  : num [1:788189] 199 44 15 123 173 98 98 211 150 268 ...
##  $ start_station_name: chr [1:788189] "Wabash Ave & Grand Ave" "State St & Randolph St" "Racine Ave &
##  $ end_station_id    : num [1:788189] 84 624 644 176 35 49 49 142 148 141 ...
##  $ end_station_name  : chr [1:788189] "Milwaukee Ave & Grand Ave" "Dearborn St & Van Buren St (*)" "
##  $ member_casual     : chr [1:788189] "Annual" "Annual" "Annual" "Annual" ...
##  $ date              : Date[1:788189], format: "2019-01-01" "2019-01-01" ...
##  $ month             : chr [1:788189] "01" "01" "01" "01" ...
##  $ day               : chr [1:788189] "01" "01" "01" "01" ...
##  $ year              : chr [1:788189] "2019" "2019" "2019" "2019" ...
##  $ day_of_week       : chr [1:788189] "Tuesday" "Tuesday" "Tuesday" "Tuesday" ...
##  $ ride_length       : 'difftime' num [1:788189] 390 441 829 1783 ...
##   ..- attr(*, "units")= chr "secs"
```

I need to make sure the column for ride_length isn't in factor form, I need this to be numeric if I wish to make calculations such as mean,median, mode later on.

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

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

Removing Bad Data

As I had previously learned with station 675 a lot of these results were either negative or had 0 min rides, only by viewing this dataframe after the processes above I was able to easily clarify that this is a HQ Quality control centre where bikes are undocked for a brief period but not ridden. (updated individual spreadsheet after this to reflect correctly)

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

Time to Anaylse the Data

Descriptive analysis on ride_length (all figures in seconds)

```
summary(all_trips_v2) # Overall Summary of the data
```

```
##    ride_id            started_at
##  Length:788189      Min.   :2019-01-01 00:04:37.00
##  Class :character   1st Qu.:2019-02-28 13:39:58.00
##  Mode  :character   Median :2020-01-07 07:59:53.00
##                     Mean   :2019-08-31 14:14:43.81
##                     3rd Qu.:2020-02-19 12:38:46.00
##                     Max.   :2020-03-31 23:51:34.00
##     ended_at                       rideable_type      start_station_id
##  Min.   :2019-01-01 00:11:07.00   Length:788189      Min.   :  2.0
##  1st Qu.:2019-02-28 13:51:45.00   Class :character   1st Qu.: 77.0
##  Median :2020-01-07 08:10:57.00   Mode  :character   Median :174.0
##  Mean   :2019-08-31 14:34:33.26                      Mean   :202.2
##  3rd Qu.:2020-02-19 12:57:45.00                      3rd Qu.:289.0
##  Max.   :2020-05-19 20:10:34.00                      Max.   :673.0
##  start_station_name end_station_id  end_station_name   member_casual
##  Length:788189      Min.   :  2.0   Length:788189      Length:788189
##  Class :character   1st Qu.: 77.0   Class :character   Class :character
##  Mode  :character   Median :173.0   Mode  :character   Mode  :character
##                     Mean   :202.1
##                     3rd Qu.:289.0
##                     Max.   :675.0
##       date                month               day                year
##  Min.   :2019-01-01   Length:788189      Length:788189      Length:788189
##  1st Qu.:2019-02-28   Class :character   Class :character   Class :character
##  Median :2020-01-07   Mode  :character   Mode  :character   Mode  :character
##  Mean   :2019-08-31
##  3rd Qu.:2020-02-19
##  Max.   :2020-03-31
##  day_of_week         ride_length
##  Length:788189      Min.   :       1
##  Class :character   1st Qu.:     331
##  Mode  :character   Median :     539
##                     Mean   :    1189
##                     3rd Qu.:     912
##                     Max.   :10632022
```

```
mean(all_trips_v2$ride_length) # Mean of the ride length
```

```
## [1] 1189.459
```

```
total_ride_length <- sum(all_trips_v2$ride_length, na.rm = TRUE)

number_of_rides <- nrow(all_trips_v2)

average_ride_length <- total_ride_length / number_of_rides

print(average_ride_length) # This chuck of code will get the Average ride length
```

```
## [1] 1189.459
```

```
median(all_trips_v2$ride_length) # Median ride length
```

```
## [1] 539
```

```r
max(all_trips_v2$ride_length) # Longest ride
```

```
## [1] 10632022
```

```r
min(all_trips_v2$ride_length) # Shortest ride
```

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

Time to compare Members and Casuals

Now it's time to see the difference using these metrics between the two userbases

```r
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual, FUN = mean) # The mean between the two
```

```
##   all_trips_v2$member_casual all_trips_v2$ride_length
## 1                     Annual                 795.2523
## 2                     Casual                5372.7839
```

```r
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual, FUN = median)
```

```
##   all_trips_v2$member_casual all_trips_v2$ride_length
## 1                     Annual                      508
## 2                     Casual                     1393
```

```r
# The Median between the two userbases
```

```r
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual, FUN = max)
```

```
##   all_trips_v2$member_casual all_trips_v2$ride_length
## 1                     Annual                  6096428
## 2                     Casual                 10632022
```

```r
# The longest ride between the two userbases
```

```r
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual, FUN = min)
```

```
##   all_trips_v2$member_casual all_trips_v2$ride_length
## 1                     Annual                        1
## 2                     Casual                        2
```

```r
# The shortest ride between the two userbases
```

```r
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                      Annual                   Friday                 796.7338
## 2                      Casual                   Friday                6090.7373
## 3                      Annual                   Monday                 822.3112
## 4                      Casual                   Monday                4752.0504
## 5                      Annual                 Saturday                 974.0730
## 6                      Casual                 Saturday                4950.7708
## 7                      Annual                   Sunday                 972.9383
## 8                      Casual                   Sunday                5061.3044
## 9                      Annual                 Thursday                 707.2093
## 10                     Casual                 Thursday                8451.6669
## 11                     Annual                  Tuesday                 769.4416
## 12                     Casual                  Tuesday                4561.8039
## 13                     Annual                Wednesday                 711.9838
## 14                     Casual                Wednesday                4480.3724
```

The days of the week are out of order making this data harder to understand I need to write the below code to fix this issues

```
all_trips_v2$day_of_week <- ordered(all_trips_v2$day_of_week, levels=c("Sunday", "Monday", "Tuesday", "
```

Now I will try to find the average ride time by userbase on a daily basis

```
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                      Annual                   Sunday                 972.9383
## 2                      Casual                   Sunday                5061.3044
## 3                      Annual                   Monday                 822.3112
## 4                      Casual                   Monday                4752.0504
## 5                      Annual                  Tuesday                 769.4416
## 6                      Casual                  Tuesday                4561.8039
## 7                      Annual                Wednesday                 711.9838
## 8                      Casual                Wednesday                4480.3724
## 9                      Annual                 Thursday                 707.2093
## 10                     Casual                 Thursday                8451.6669
## 11                     Annual                   Friday                 796.7338
## 12                     Casual                   Friday                6090.7373
## 13                     Annual                 Saturday                 974.0730
## 14                     Casual                 Saturday                4950.7708
```

This looks much clearer to read and understand.

Lets Analyse the Userbase by Weekday

```
all_trips_v2 %>%
  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()                              #calculates the number of rides and average
            ,average_duration = mean(ride_length)) %>%      # calculates the average duration
  arrange(member_casual, weekday)   # sorts the data
```

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

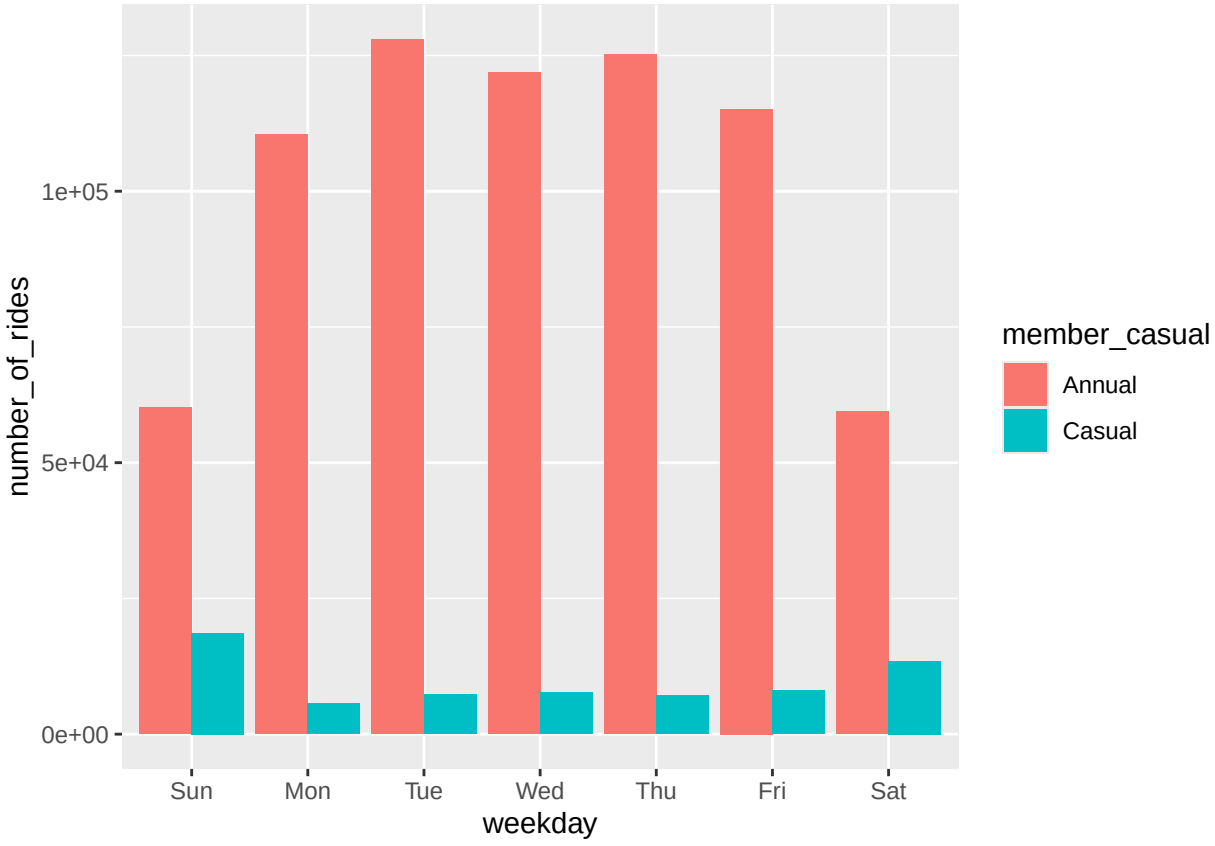
```
## # A tibble: 14 x 4
## # Groups:   member_casual [2]
##    member_casual weekday number_of_rides average_duration
##    <chr>         <ord>             <int>            <dbl>
##  1 Annual        Sun               60197             973.
##  2 Annual        Mon              110430             822.
##  3 Annual        Tue              127974             769.
##  4 Annual        Wed              121902             712.
##  5 Annual        Thu              125228             707.
##  6 Annual        Fri              115168             797.
##  7 Annual        Sat               59413             974.
##  8 Casual        Sun               18652            5061.
##  9 Casual        Mon                5591            4752.
## 10 Casual        Tue                7311            4562.
## 11 Casual        Wed                7690            4480.
## 12 Casual        Thu                7147            8452.
## 13 Casual        Fri                8013            6091.
## 14 Casual        Sat               13473            4951.
```

Time to Visualise this

The Below Graph shows the number of ride by userbase

```
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")
```

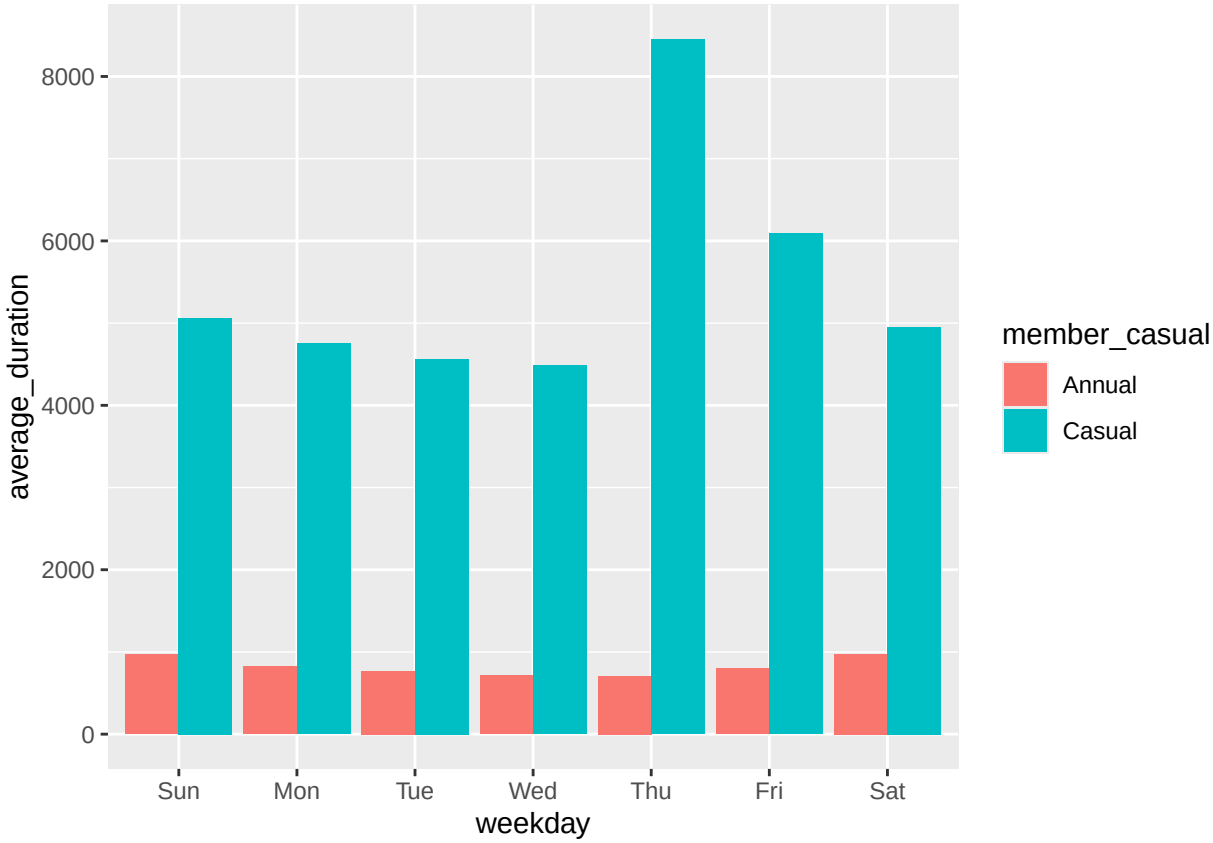
```
## `summarise()` has grouped output by 'member_casual'. You can override using the
## `.groups` argument.
```



The Below graph shows the average duration by userbase

```
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.
```



Summary

The two Graphs currently displayed show the despite the Annual Users using the service more regularly with there highest uptake being in the working week, despite this the duration of the trips is on average lower than a casual user during this time.

This is most likely due to the trips took by annual members are commutes to work or to events.

The casual users don't use the service through the week but there is an uptake of rides on weekends, overall the rides a casual user does is longer than a Annual rider. this is most likely less frequent but long bike rides on a weekend when they are off work.

Suggestions Based on this

For an Weekend only annual membership. this may have get casual users who only use the service on a weekend to be more inclined to a subscription model.

Monthly or Bi Monthly passes with a set number of hours linked to the pass i.e 1 Month 35 Hours, 2 Months 80 Hours. This may get causal users who will use the service for long durations of time on a weekend to pay upfront but not have the fear of being in an annual subscription model initially. this may In turn make them use the service during the week which may make them reconsider an annual subscription.

Increased marketing towards the idea of using the service for short regular journeys i.e going to nearby social events / commuting to work.

Further Visualisation / Analysis

I am going to create a CSV file with this dataframe that I then perform more visualisations using tableau https://public.tableau.com/app/profile/ryan.welsh6016/vizzes