Google Cyclistic Capstone Project

Nyerovwo Victor Ataro

2022-10-03

About the Project.

This project is about the Cyclistic bike-share analysis case study! for a fictional company, Cyclistic. The bike-share program consist of over 5,800 bicycles and 600 docking stations and the company have two customer base casual and Annual members. The data is collected directly by Motivate, Inc.

Business task

Design marketing strategies aimed at converting casual riders into annual members.

Objectives

- How do annual members and casual riders use Cyclistic bikes differently?
- Why would casual riders buy Cyclistic annual memberships?
- How can Cyclistic use digital media to influence casual riders to become members?

Process Data

Due to the large dataset, I used RStudio Desktop to analyse, clean, summarize and visualize the data for this project. Data was examined to gain a general understanding of field content, data formats, and data integrity. Checking column names throughout the 12 original files and looking for missing values, trailing white spaces, duplicate records, and other data abnormalities were all part of the data assessment.

Install the packages.

```
install.packages("tidyverse")
```

```
## package 'tidyverse' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Nyerovwo\AppData\Local\Temp\RtmpE7FDwH\downloaded_packages
```

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

```
## package 'ggplot2' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Nyerovwo\AppData\Local\Temp\RtmpE7FDwH\downloaded_packages
```

```
install.packages("dplyr")
```

```
## package 'dplyr' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Nyerovwo\AppData\Local\Temp\RtmpE7FDwH\downloaded_packages
```

```
install.packages("janitor")
```

```
## package 'janitor' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Nyerovwo\AppData\Local\Temp\RtmpE7FDwH\downloaded_packages
```

```
install.packages("lubridate")
```

```
## package 'lubridate' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Nyerovwo\AppData\Local\Temp\RtmpE7FDwH\downloaded_packages
```

```
install.packages("skimr")
```

```
## package 'skimr' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Nyerovwo\AppData\Local\Temp\RtmpE7FDwH\downloaded_packages
```

```
install.packages("here")
```

```
## package 'here' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Nyerovwo\AppData\Local\Temp\RtmpE7FDwH\downloaded_packages
```

Loading the Data library

Once a package is installed, you can load it by running the `library()` function with the package name inside the parentheses:

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag

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

```
library(lubridate)
```

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

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

```
library(janitor)
```

```
##
## Attaching package: 'janitor'

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

```
library(tidyr)
library(here)
```

```
## here() starts at F:/capstone files
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --

## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.8     v stringr 1.4.1
## v readr   2.1.2     v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x lubridate::as.difftime() masks base::as.difftime()
## x lubridate::date()        masks base::date()
## x dplyr::filter()          masks stats::filter()
## x lubridate::intersect()   masks base::intersect()
## x dplyr::lag()             masks stats::lag()
## x lubridate::setdiff()     masks base::setdiff()
## x lubridate::union()       masks base::union()
```

```
library(skimr)
```

Import data

We will use the `read_csv()` function to import data from a .csv folder

```
 Jan <- read.csv("202101-divvy-tripdata.csv")
Feb <- read.csv("202102-divvy-tripdata.csv")
Mar <- read.csv("202103-divvy-tripdata.csv")
Apr <- read.csv("202004-divvy-tripdata.csv")
May<- read.csv("202005-divvy-tripdata.csv")
Jun <- read.csv("202006-divvy-tripdata.csv")
Jul <- read.csv("202007-divvy-tripdata.csv")
Aug <- read.csv("202008-divvy-tripdata.csv")
Sep <- read.csv("202009-divvy-tripdata.csv")
Oct <- read.csv("202010-divvy-tripdata.csv")
Nov <- read.csv("202011-divvy-tripdata.csv")
Dec <- read.csv("202012-divvy-tripdata.csv")
```

Merging data into a data frame

```
data <- rbind(Jan,Feb,Mar,Apr,May,Jun,Jul,Aug,Sep,Oct,Nov,Dec)
```

Inspect the data

One common function you can use to preview the data is the `head()` function, which returns the columns and the first several rows of data

```
head(data)
```

```
##            ride_id rideable_type          started_at            ended_at
## 1 E19E6F1B8D4C42ED electric_bike 2021-01-23 16:14:19 2021-01-23 16:24:44
## 2 DC88F20C2C55F27F electric_bike 2021-01-27 18:43:08 2021-01-27 18:47:12
## 3 EC45C94683FE3F27 electric_bike 2021-01-21 22:35:54 2021-01-21 22:37:14
## 4 4FA453A75AE377DB electric_bike 2021-01-07 13:31:13 2021-01-07 13:42:55
## 5 BE5E8EB4E7263A0B electric_bike 2021-01-23 02:24:02 2021-01-23 02:24:45
## 6 5D8969F88C773979 electric_bike 2021-01-09 14:24:07 2021-01-09 15:17:54
##           start_station_name start_station_id end_station_name end_station_id
## 1 California Ave & Cortez St            17660
## 2 California Ave & Cortez St            17660
## 3 California Ave & Cortez St            17660
## 4 California Ave & Cortez St            17660
## 5 California Ave & Cortez St            17660
## 6 California Ave & Cortez St            17660
##   start_lat start_lng end_lat end_lng member_casual
## 1  41.90034 -87.69674   41.89  -87.72        member
## 2  41.90033 -87.69671   41.90  -87.69        member
## 3  41.90031 -87.69664   41.90  -87.70        member
```

```
## 4  41.90040 -87.69666   41.92  -87.69        member
## 5  41.90033 -87.69670   41.90  -87.70        casual
## 6  41.90041 -87.69676   41.94  -87.71        casual
```

In addition to `head()`, there are a number of other useful functions to summarize or preview your data. For example, the `str()`, and `glimpse()` functions will both provide summaries of each column in your data arranged horizontally.

```
str(data)
```

```
## 'data.frame':    3489748 obs. of  13 variables:
##  $ ride_id           : chr  "E19E6F1B8D4C42ED" "DC88F20C2C55F27F" "EC45C94683FE3F27" "4FA453A75AE377
##  $ rideable_type     : chr  "electric_bike" "electric_bike" "electric_bike" "electric_bike" ...
##  $ started_at        : chr  "2021-01-23 16:14:19" "2021-01-27 18:43:08" "2021-01-21 22:35:54" "2021-
##  $ ended_at          : chr  "2021-01-23 16:24:44" "2021-01-27 18:47:12" "2021-01-21 22:37:14" "2021-
##  $ start_station_name: chr  "California Ave & Cortez St" "California Ave & Cortez St" "California Av
##  $ start_station_id  : chr  "17660" "17660" "17660" "17660" ...
##  $ end_station_name  : chr  "" "" "" "" ...
##  $ end_station_id    : chr  "" "" "" "" ...
##  $ start_lat         : num  41.9 41.9 41.9 41.9 41.9 ...
##  $ start_lng         : num  -87.7 -87.7 -87.7 -87.7 -87.7 ...
##  $ end_lat           : num  41.9 41.9 41.9 41.9 41.9 ...
##  $ end_lng           : num  -87.7 -87.7 -87.7 -87.7 -87.7 ...
##  $ member_casual     : chr  "member" "member" "member" "member" ...
```

You can also use `colnames()` to get a list the column names in your data set.

```
colnames(data)
```

```
##  [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"
```

```
glimpse(data)
```

```
## Rows: 3,489,748
## Columns: 13
## $ ride_id            <chr> "E19E6F1B8D4C42ED", "DC88F20C2C55F27F", "EC45C94683~
## $ rideable_type      <chr> "electric_bike", "electric_bike", "electric_bike", ~
## $ started_at         <chr> "2021-01-23 16:14:19", "2021-01-27 18:43:08", "2021~
## $ ended_at           <chr> "2021-01-23 16:24:44", "2021-01-27 18:47:12", "2021~
## $ start_station_name <chr> "California Ave & Cortez St", "California Ave & Cor~
## $ start_station_id   <chr> "17660", "17660", "17660", "17660", "17660", "17660~
## $ end_station_name   <chr> "", "", "", "", "", "", "", "", "", "Wood St & Augu~
## $ end_station_id     <chr> "", "", "", "", "", "", "", "", "", "657", "13258",~
## $ start_lat          <dbl> 41.90034, 41.90033, 41.90031, 41.90040, 41.90033, 4~
```

```
## $ start_lng          <dbl> -87.69674, -87.69671, -87.69664, -87.69666, -87.696~
## $ end_lat            <dbl> 41.89000, 41.90000, 41.90000, 41.92000, 41.90000, 4~
## $ end_lng            <dbl> -87.72000, -87.69000, -87.70000, -87.69000, -87.700~
## $ member_casual      <chr> "member", "member", "member", "member", "casual", "~
```

Data Cleaning

Removed empty columns and rows and cleaned variable names in one

```
data_remove_row <- data %>%
  remove_empty(which = c("cols", "rows")) %>%
  clean_names()
```

The following variables have been transformed from character to numeric variables

```
data2 <- data_remove_row
```

```
data_1 <- data2 %>%
  mutate(
    start_lat = as.numeric(start_lat),
    start_lng = as.numeric(start_lng),
    end_lat = as.numeric(end_lat),
    end_lng = as.numeric(end_lng)
  )
```

Let view the data

```
str(data_1)
```

```
## 'data.frame':    3489748 obs. of  13 variables:
##  $ ride_id           : chr  "E19E6F1B8D4C42ED" "DC88F20C2C55F27F" "EC45C94683FE3F27" "4FA453A75AE377
##  $ rideable_type     : chr  "electric_bike" "electric_bike" "electric_bike" "electric_bike" ...
##  $ started_at        : chr  "2021-01-23 16:14:19" "2021-01-27 18:43:08" "2021-01-21 22:35:54" "2021-
##  $ ended_at          : chr  "2021-01-23 16:24:44" "2021-01-27 18:47:12" "2021-01-21 22:37:14" "2021-
##  $ start_station_name: chr  "California Ave & Cortez St" "California Ave & Cortez St" "California Av
##  $ start_station_id  : chr  "17660" "17660" "17660" "17660" ...
##  $ end_station_name  : chr  "" "" "" "" ...
##  $ end_station_id    : chr  "" "" "" "" ...
##  $ start_lat         : num  41.9 41.9 41.9 41.9 41.9 ...
##  $ start_lng         : num  -87.7 -87.7 -87.7 -87.7 -87.7 ...
##  $ end_lat           : num  41.9 41.9 41.9 41.9 41.9 ...
##  $ end_lng           : num  -87.7 -87.7 -87.7 -87.7 -87.7 ...
##  $ member_casual     : chr  "member" "member" "member" "member" ...
```

Convert to date-time coglimpse(data_1)lumns

```
data_2 <- data_1 %>%
  mutate(
    started_at = ymd_hms(as_datetime(started_at)),
    ended_at = ymd_hms(as_datetime(ended_at))
  )
```

To get the summary overview

```
glimpse(data_2)
```

```
## Rows: 3,489,748
## Columns: 13
## $ ride_id            <chr> "E19E6F1B8D4C42ED", "DC88F20C2C55F27F", "EC45C94683~
## $ rideable_type      <chr> "electric_bike", "electric_bike", "electric_bike", ~
## $ started_at         <dttm> 2021-01-23 16:14:19, 2021-01-27 18:43:08, 2021-01-~
## $ ended_at           <dttm> 2021-01-23 16:24:44, 2021-01-27 18:47:12, 2021-01-~
## $ start_station_name <chr> "California Ave & Cortez St", "California Ave & Cor~
## $ start_station_id   <chr> "17660", "17660", "17660", "17660", "17660", "17660~
## $ end_station_name   <chr> "", "", "", "", "", "", "", "", "", "Wood St & Augu~
## $ end_station_id     <chr> "", "", "", "", "", "", "", "", "", "657", "13258",~
## $ start_lat          <dbl> 41.90034, 41.90033, 41.90031, 41.90040, 41.90033, 4~
## $ start_lng          <dbl> -87.69674, -87.69671, -87.69664, -87.69666, -87.696~
## $ end_lat            <dbl> 41.89000, 41.90000, 41.90000, 41.92000, 41.90000, 4~
## $ end_lng            <dbl> -87.72000, -87.69000, -87.70000, -87.69000, -87.700~
## $ member_casual      <chr> "member", "member", "member", "member", "casual", "~
```

Define a new columns based on the started_at, date-time column and do some conversion on date-time and time different between started_at and ende_at

```
data_3 <- data_2 %>%
  mutate(
    hour_start = hour(started_at),
    weekday = wday(started_at, label = TRUE, abbr = FALSE),
    month = month(started_at, label = TRUE, abbr =FALSE),
    day = day(started_at),
    week = strftime(started_at, format = "%V"),
    trip_time = difftime(ended_at, started_at, units = "mins")
  )
```

Creat column rideable_types and types of Bikes and let it be in an ordered form.

```
data_4 <- data_3 %>%
  mutate(
    rideable_type = recode(as_factor(rideable_type),
```

```
                            "classic_bike" = "classic",
                            "electric_bike"= "electric",
                            "docked_bike" = "docked"),
    member_casual = as_factor(member_casual)
  )
```

Rename columns rideable_type and member_casual to Bikes and Users

```
data_4 <- data_4 %>%
  rename(
    bikes = rideable_type,
    users = member_casual
  )
```

Create a new data frame that contain trip_time of 1 min and less than or equal to 24 h = 1440 min

```
data_5 <- data_4 %>%
  filter(trip_time > 0 & trip_time <= 1440)
```

TO get the comprehensive summary of the data set let use skim_without_charts

```
data_5 %>%
  skim_without_charts()
```

Table 1: Data summary

Name	Piped data
Number of rows	3475928
Number of columns	19
Column type frequency:	
character	6
difftime	1
factor	4
numeric	6
POSIXct	2
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
ride_id	0	1.00	16	16	0	3475928	0

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
start_station_name	0	1.00	0	53	122126	709	0
start_station_id	83576	0.98	0	35	39176	1260	0
end_station_name	0	1.00	0	53	142362	707	0
end_station_id	97513	0.97	0	35	45310	1260	0
week	0	1.00	2	2	0	53	0

Variable type: difftime

skim_variable	n_missing	complete_rate	min	max	median	n_unique
trip_time	0	1	0.02 mins	1439.9 mins	14.55 mins	23917

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
bikes	0	1	FALSE	3	doc: 2545174, ele: 611119, cla: 319635
users	0	1	FALSE	2	mem: 2051510, cas: 1424418
weekday	0	1	TRUE	7	Sat: 657517, Sun: 526954, Fri: 514476, Thu: 466445
month	0	1	TRUE	12	Aug: 618986, Jul: 548954, Sep: 530456, Oct: 386497

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
start_lat	0	1	41.90	0.04	41.64	41.88	41.90	41.93	42.08
start_lng	0	1	-87.64	0.03	-87.87	-87.66	-87.64	-87.63	-87.52
end_lat	3993	1	41.90	0.04	41.54	41.88	41.90	41.93	42.16
end_lng	3993	1	-87.65	0.03	-88.07	-87.66	-87.64	-87.63	-87.44
hour_start	0	1	14.31	4.60	0.00	11.00	15.00	18.00	23.00
day	0	1	15.86	8.76	1.00	8.00	16.00	24.00	31.00

Variable type: POSIXct

skim_variable	n_missing	complete_rate	min	max	median	n_unique
started_at	0	1	2020-04-01 00:00:30	2021-03-31 23:59:08	2020-08-29 14:48:54	3032959
ended_at	0	1	2020-04-01 00:10:45	2021-04-01 08:45:23	2020-08-29 15:18:42	3017771

To get structure of the datafram

```
str(data_5)
```

```
## 'data.frame':    3475928 obs. of  19 variables:
##  $ ride_id           : chr  "E19E6F1B8D4C42ED" "DC88F20C2C55F27F" "EC45C94683FE3F27" "4FA453A75AE377
##  $ bikes             : Factor w/ 3 levels "electric","classic",..: 1 1 1 1 1 1 1 1 1 2 ...
##  $ started_at        : POSIXct, format: "2021-01-23 16:14:19" "2021-01-27 18:43:08" ...
##  $ ended_at          : POSIXct, format: "2021-01-23 16:24:44" "2021-01-27 18:47:12" ...
##  $ start_station_name: chr  "California Ave & Cortez St" "California Ave & Cortez St" "California Av
##  $ start_station_id  : chr  "17660" "17660" "17660" "17660" ...
##  $ end_station_name  : chr  "" "" "" "" ...
##  $ end_station_id    : chr  "" "" "" "" ...
##  $ start_lat         : num  41.9 41.9 41.9 41.9 41.9 ...
##  $ start_lng         : num  -87.7 -87.7 -87.7 -87.7 -87.7 ...
##  $ end_lat           : num  41.9 41.9 41.9 41.9 41.9 ...
##  $ end_lng           : num  -87.7 -87.7 -87.7 -87.7 -87.7 ...
##  $ users             : Factor w/ 2 levels "member","casual": 1 1 1 1 2 2 1 1 1 1 ...
##  $ hour_start        : int  16 18 22 13 2 14 5 15 9 19 ...
##  $ weekday           : Ord.factor w/ 7 levels "Sunday"<"Monday"<..: 7 4 5 5 7 7 2 5 7 1 ...
##  $ month             : Ord.factor w/ 12 levels "January"<"February"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ day               : int  23 27 21 7 23 9 4 14 9 24 ...
##  $ week              : chr  "03" "04" "03" "01" ...
##  $ trip_time         : 'difftime' num  10.4166666666667 4.06666666666667 1.33333333333333 11.7 ...
##   ..- attr(*, "units")= chr "mins"
```

Let view the columns and the first several rows of the data

```
head(data_5)
```

```
##            ride_id    bikes          started_at            ended_at
## 1 E19E6F1B8D4C42ED electric 2021-01-23 16:14:19 2021-01-23 16:24:44
## 2 DC88F20C2C55F27F electric 2021-01-27 18:43:08 2021-01-27 18:47:12
## 3 EC45C94683FE3F27 electric 2021-01-21 22:35:54 2021-01-21 22:37:14
## 4 4FA453A75AE377DB electric 2021-01-07 13:31:13 2021-01-07 13:42:55
## 5 BE5E8EB4E7263A0B electric 2021-01-23 02:24:02 2021-01-23 02:24:45
## 6 5D8969F88C773979 electric 2021-01-09 14:24:07 2021-01-09 15:17:54
##           start_station_name start_station_id end_station_name end_station_id
## 1 California Ave & Cortez St            17660
## 2 California Ave & Cortez St            17660
## 3 California Ave & Cortez St            17660
## 4 California Ave & Cortez St            17660
## 5 California Ave & Cortez St            17660
## 6 California Ave & Cortez St            17660
##   start_lat start_lng end_lat end_lng  users hour_start   weekday   month day
## 1  41.90034 -87.69674   41.89  -87.72 member         16  Saturday January  23
## 2  41.90033 -87.69671   41.90  -87.69 member         18 Wednesday January  27
## 3  41.90031 -87.69664   41.90  -87.70 member         22  Thursday January  21
## 4  41.90040 -87.69666   41.92  -87.69 member         13  Thursday January   7
## 5  41.90033 -87.69670   41.90  -87.70 casual          2  Saturday January  23
## 6  41.90041 -87.69676   41.94  -87.71 casual         14  Saturday January   9
##   week       trip_time
## 1   03 10.4166667 mins
## 2   04  4.0666667 mins
## 3   03  1.3333333 mins
## 4   01 11.7000000 mins
```

```
## 5   03  0.7166667 mins
## 6   01 53.7833333 mins
```

Let Create a new data frame with time variables

```
data_time = data_5 %>%
  drop_na(
    end_lat, end_lng
  ) %>%
  select(
    ride_id, users, bikes, hour_start, weekday, month, day, week, trip_time
  )
```

check if NA exist

```
colSums(is.na(data_time))
```

```
##    ride_id      users      bikes hour_start    weekday      month        day
##          0          0          0          0          0          0          0
##       week  trip_time
##          0          0
```

Create a data frame with location variables

```
data_location_df = data_5 %>%
  select(
    ride_id, start_station_name, end_station_name, start_lat, start_lng,
    end_lat, end_lng, users, trip_time
  ) %>%
  drop_na(
    start_station_name, end_station_name
  )
```

Check for NA

```
colSums(is.na(data_location_df))
```

```
##            ride_id start_station_name   end_station_name          start_lat
##                  0                  0                  0                  0
##          start_lng            end_lat            end_lng              users
##                  0               3993               3993                  0
##          trip_time
##                  0
```

Let remove the na in end_lat and end_lng

```
data_na <- data_location_df %>% filter(is.na(end_lat)==FALSE)
```

Check for na

```
colSums(is.na(data_na))
```

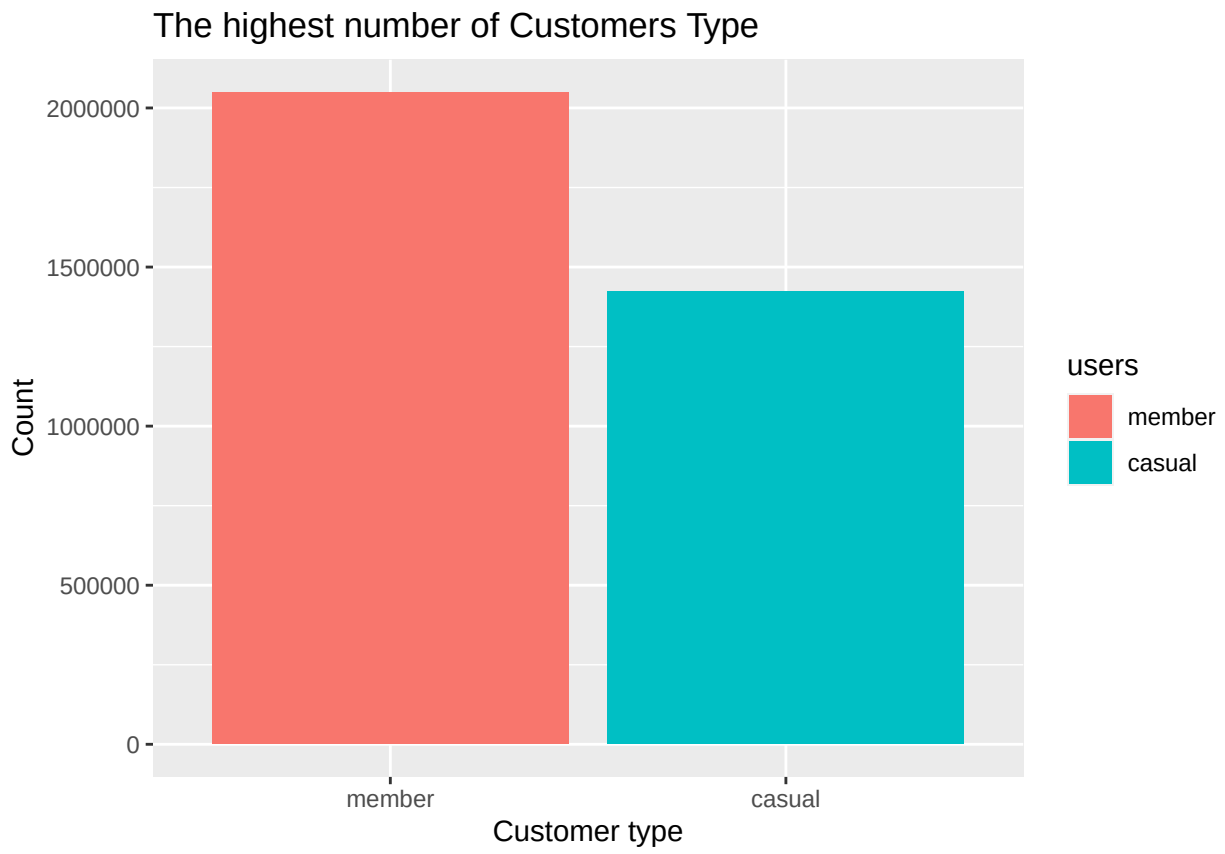
```
##            ride_id start_station_name   end_station_name          start_lat
##                  0                  0                  0                  0
##          start_lng            end_lat            end_lng              users
##                  0                  0                  0                  0
##          trip_time
##                  0
```

let name our clean data

```
data_clean <- data_na
```

To know the number of customers

```
ggplot(data = data_clean) + geom_bar(mapping = aes(x = users, fill = users)) + labs(title = "The highes
```



From the above chart it shows that we have more Annual member than casual.

graph perparation

```
newtheme <- theme_light() +
  theme(plot.title = element_text(color = "#002949", face = 'bold', size =12),
        plot.subtitle = element_text(color = "#890000", size = 10),
        plot.caption = element_text(color = '#890000', face = 'italic', size =8),
        panel.border = element_rect(color = "#002949", size = 1),
        legend.position = "right",
        legend.text = element_text(colour="blue", size=10, face="bold"),
        legend.title = element_text(colour="blue", size=10, face="bold"),
        #legend.position='none',
        axis.title.x = element_text(colour = "#890000"),
        axis.title.y = element_text(colour = "#002949"),
        axis.text.x = element_text(angle = 45, hjust = 1, color = '#890000'),
        axis.text.y = element_text(angle = 45, hjust = 1, color = '#002949'),
        axis.line = element_line(color = "#002949", size =1),
  )
```

Data Visualisation

In the Analyze phase, we must identify trends and relationships and determine how the findings and insights will help answer our business questions. As a junior data analyst, I have been tasked with the following duties: Aggregate the data so that it is useful and accessible. Organize and format your data. Perform calculations. Identify trends and relationships. R was used for the Analyze phase. This phase requires a summary of our analysis.

Analysis and Visualization of hours used in each day

Let creat a column for num_rides, average_trip, total trip grouped by users type and start hour

```
hourly_ride <-
  data_time %>%
  group_by(
    users, hour_start
  ) %>%
  summarise(
    num_rides = n(),
    average_trip = mean(trip_time),
    total_trip = sum(trip_time)
  )
```

```
## Warning in readRDS(responseFile): invalid or incomplete compressed data
```

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

Creating a chart on number of trips per hours and the types of users

```
hourly_ride %>%
  ggplot(aes(hour_start, num_rides, fill = users))+
  geom_col(position = "dodge")+
  scale_y_continuous()+
  labs(
    title = "Number of Trips per Hour",
    subtitle = "Number of trips for each hour used by users",
    caption = "Figure 1",
    x = "hour of the day",
    y = "number of rides",
  )+
  theme()
```

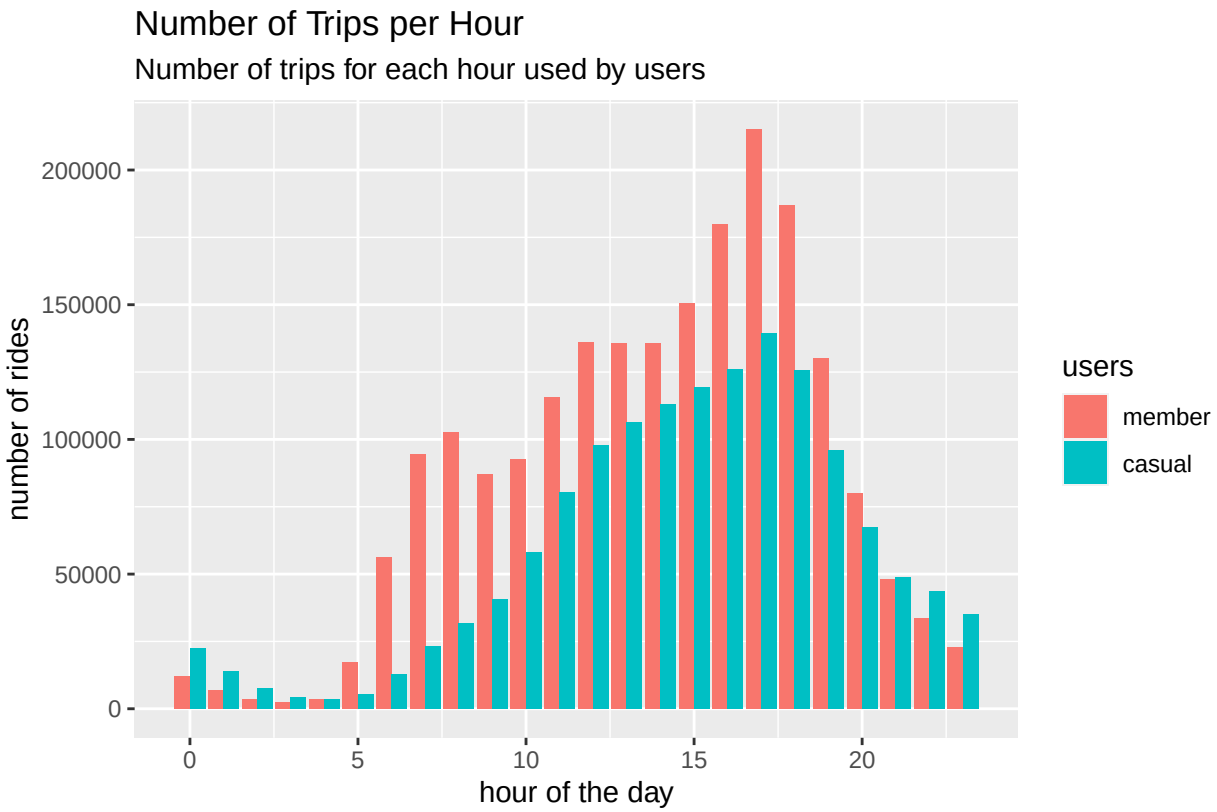


Figure 1

```
hourly_ride %>%
  ggplot(aes(hour_start, average_trip, fill = users))+
  geom_col(position = "dodge")+
  scale_y_continuous()+
  labs(
    title = "Average Number of Trips per Hour",
    subtitle = "Average number of trips for each  hour used by users",
    caption = "Figure 2",
    x = "hour of the day",
    y = "average trips duration",
  )+
  theme()
```

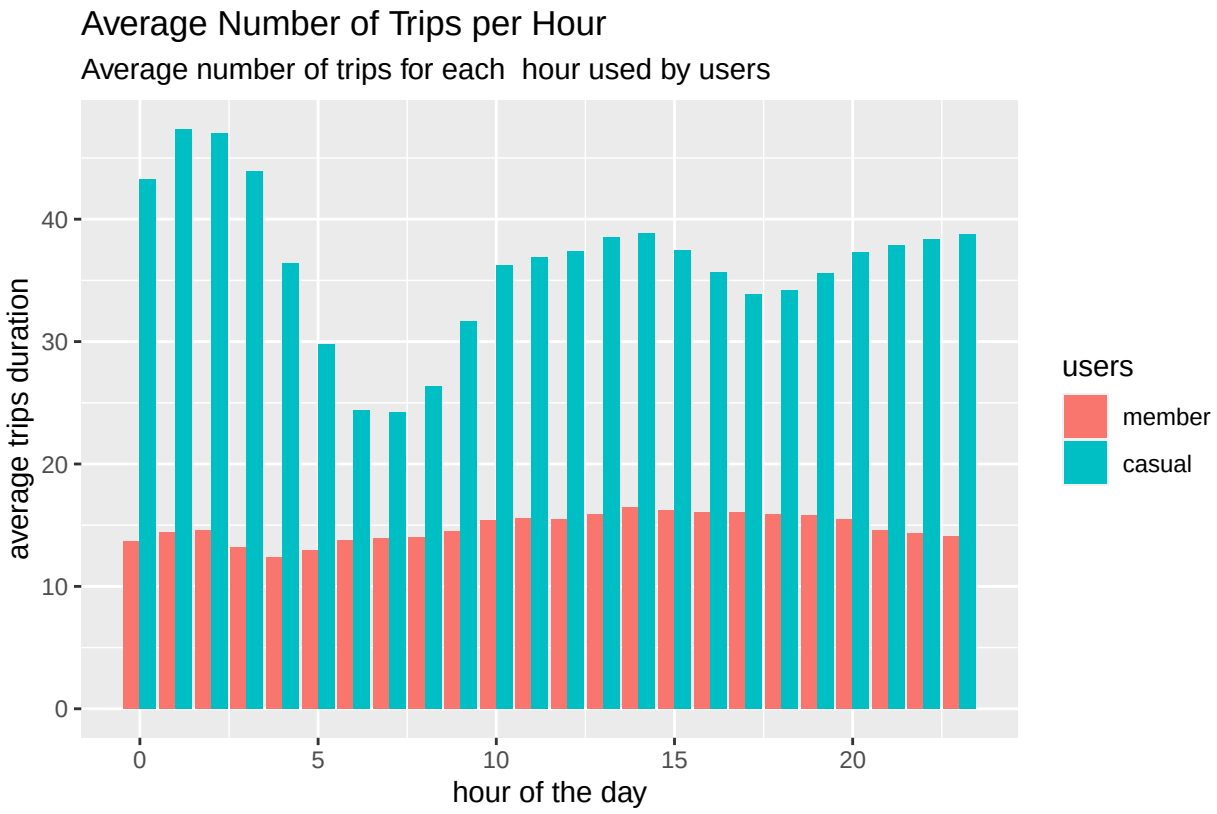


Figure 2

Chart to know the total trip per hour

```
hourly_ride %>%
  ggplot(aes(hour_start, total_trip, fill = users))+
  geom_col(show.legend = TRUE, position = "dodge")+
  scale_y_continuous()+
  labs(
    title = "Total trip Duration per Hour",
    subtitle = "Total trip duration for each hour segmented by users",
    caption = "Figure 3",
    x = "hour of the day",
    y = "total duration",
  )+
  theme()
```

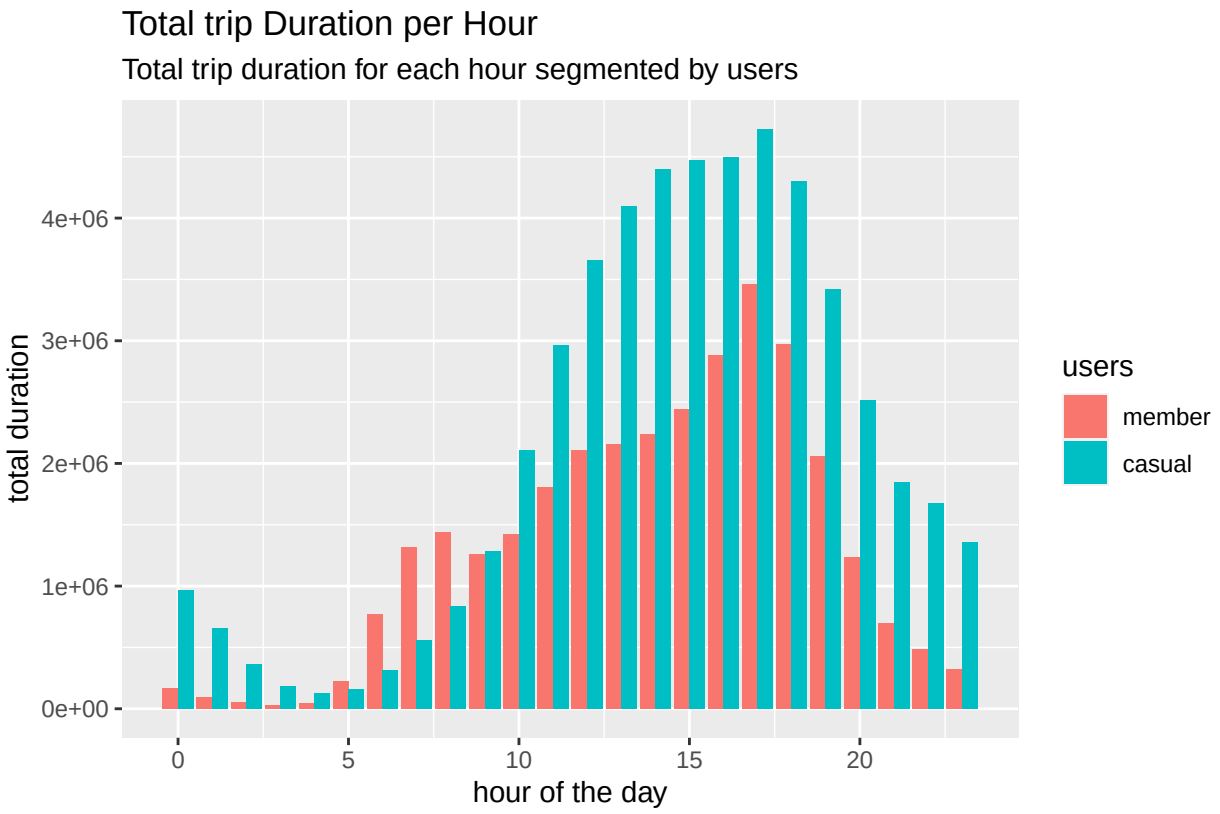


Figure 3

Hourly used finding From our hourly Analysis graph above it show that casual have the higest duration of hour, while member only have the highest between 6AM TO 8am, and the busiest time is between 9AM to 18PM.

Analysis base on days of the week

Descriptive analysis through aggregation Create new summary column on num_rides_week, average_trip_week, total_duration_week grouped by users type and week.

```
weekly_ride <- data_time %>%
  group_by(
    users, weekday
  ) %>%
  summarise(
    num_rides_week = n(),
    avg_rides_week = mean(trip_time),
    total_duration_week = sum(trip_time)
  )
```

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

Number of trips per week days

```
weekly_ride %>%
  ggplot(aes(weekday, num_rides_week, fill = users))+
    geom_col(position = "dodge")+
    scale_y_continuous()+
    labs(
      title = "Number of trips in days of the Week used by Users",
      subtitle = "Number of trips for each day of the week",
      caption = "Fig 4",
      x = "day of the week",
      y = "number of trips"
    )+
    theme()
```

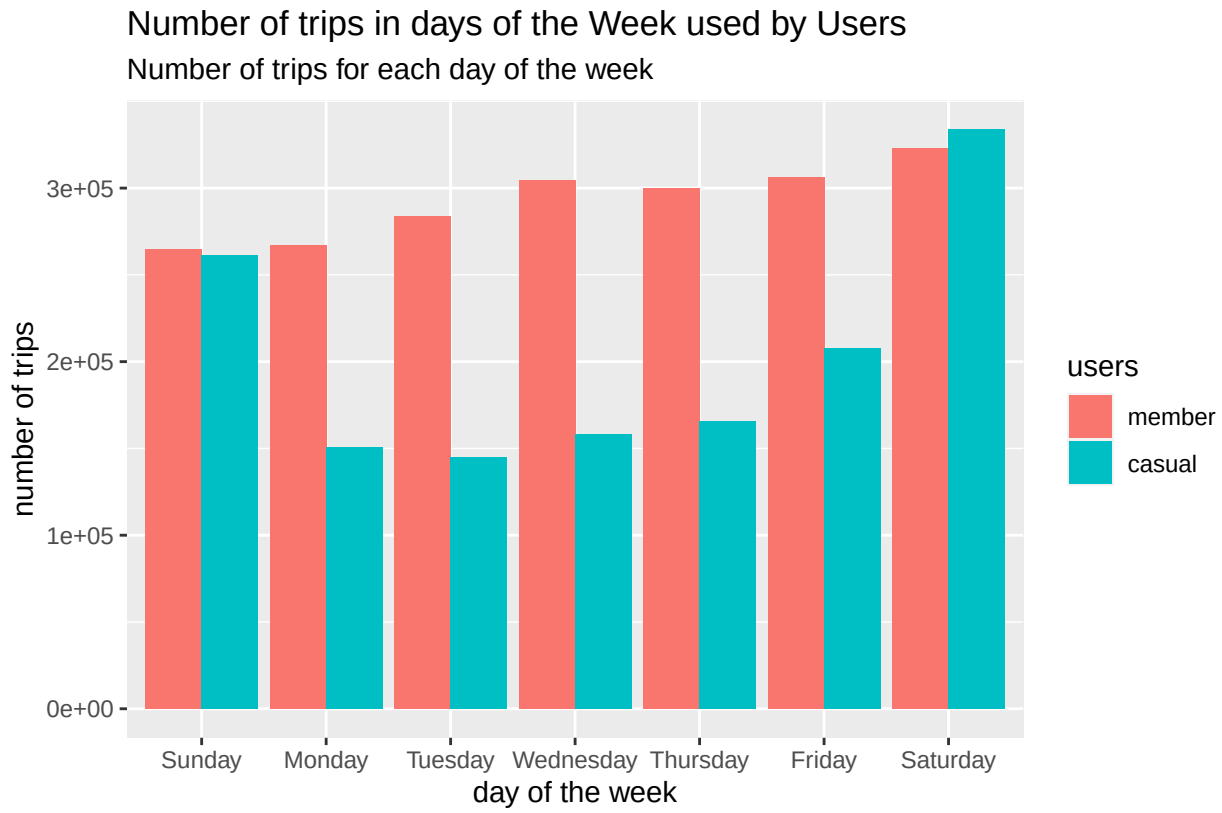


Fig 4

Average number of trips per the week days

```
weekly_ride %>%
  ggplot(aes(weekday, avg_rides_week, fill = users))+
  geom_col(position = "dodge") +
  scale_y_continuous() +
  labs(
```

```
    title = "Average Trips by Week Days used by Users",
    subtitle = "Average Number of trips for each day of the week",
    caption = "Fig 5",
    x = "day of the week",
    y = " avg number of trips"
  )+
  theme()
```

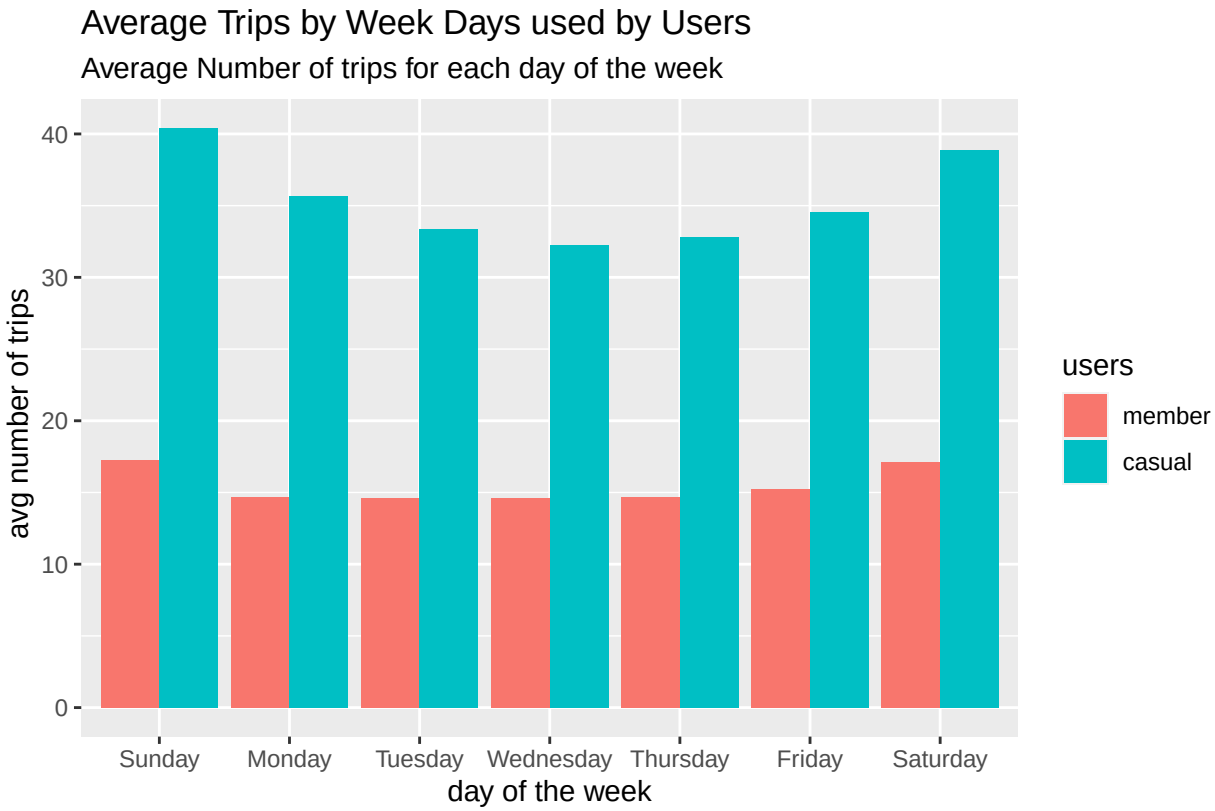


Fig 5

The total trip for each days of the week

```
weekly_ride %>%
  ggplot(aes(weekday, total_duration_week, fill = users))+
  geom_col(position = "dodge")+
  scale_y_continuous() +
  labs(
    title = "The total trip time for each days of the week used by Users",
    subtitle = "Total Trips for each day of the week",
    caption = "Fig 6",
    x = "day of the week",
    y = " total time trips"
  )+
  theme()
```

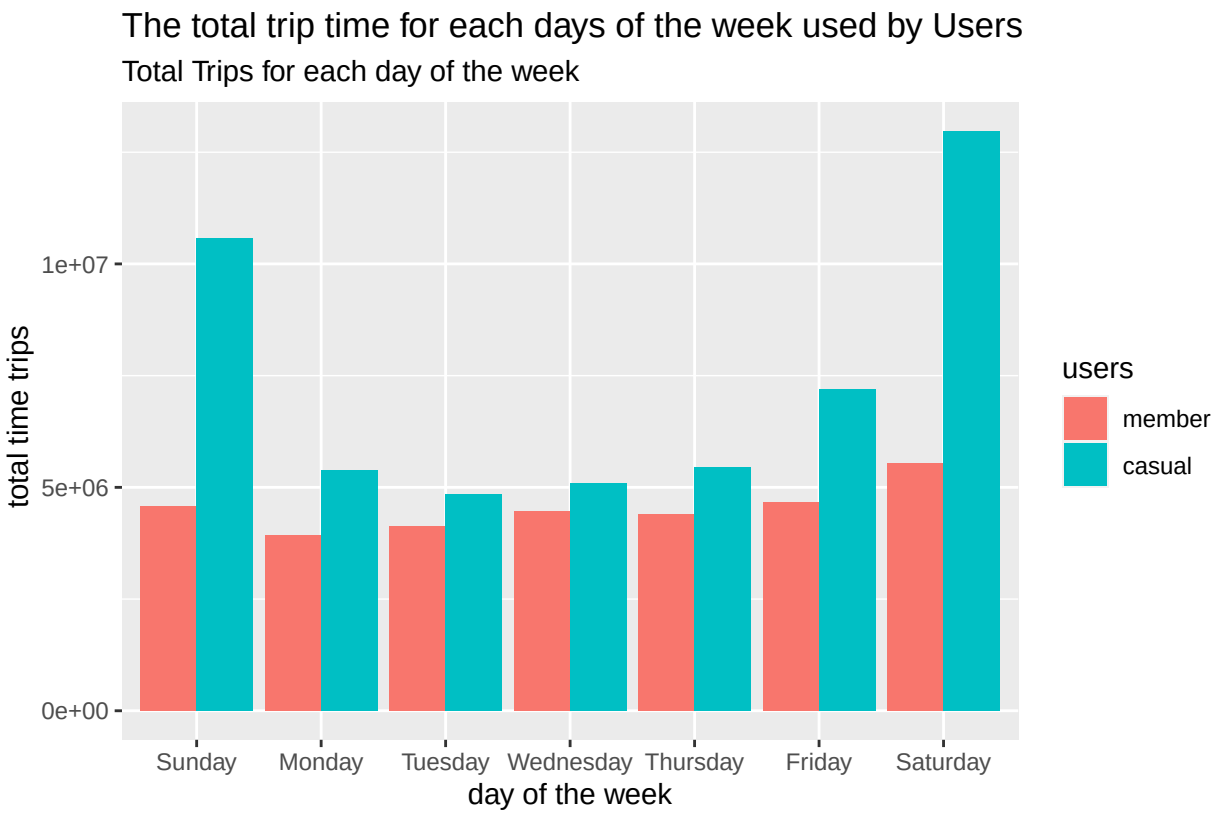


Fig 6

Week days finding

From the week days graph, it show that Casual customers have the highest trips and Saturday and Sunday recorded the highest trip time.

Analysis for trips during each month and the time the trips took place

Descriptive analysis through aggregation

```
monthly_ride <- data_time %>%
  group_by(
    users, month
  ) %>%
  summarise(
    nr_rides_month = n(),
    avg_rides_month = mean(trip_time),
    total_time_month = sum(trip_time)
  )
```

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

Number of trips by month

```
monthly_ride %>%
  ggplot(aes(month, nr_rides_month, fill = users))+
  geom_col(position = "dodge")+
  scale_y_continuous()+
  labs(
    title = "Number of Trips by Month used by Users",
    subtitle = "Number Trips for each Month",
    caption = "Fig 7",
    x = "month",
    y = " number of trips"
  )+
  theme()
```

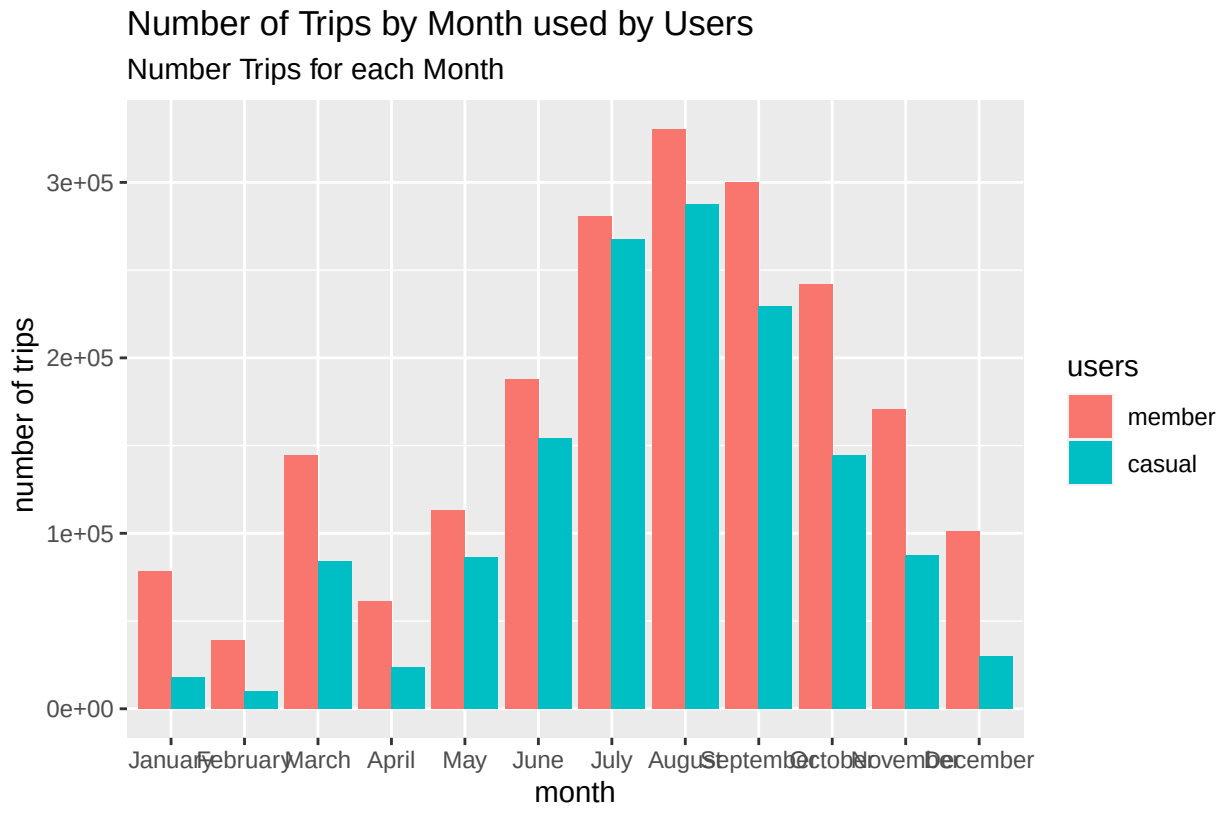


Fig 7

Number of trip pert month From the graph above it show that members are consistent all through the year and they have the highest number of trip between June to October.

Average trips for each month

```
monthly_ride %>%
  ggplot(aes(month, avg_rides_month, fill = users))+
  geom_col(position = "dodge")+
```

```
scale_y_continuous()+
labs(
  title = "Average Trips by Month used by Users",
  subtitle = "Average Trips for each Month",
  caption = "Fig 8",
  x = "month",
  y = "average trips time"
)+
theme()
```

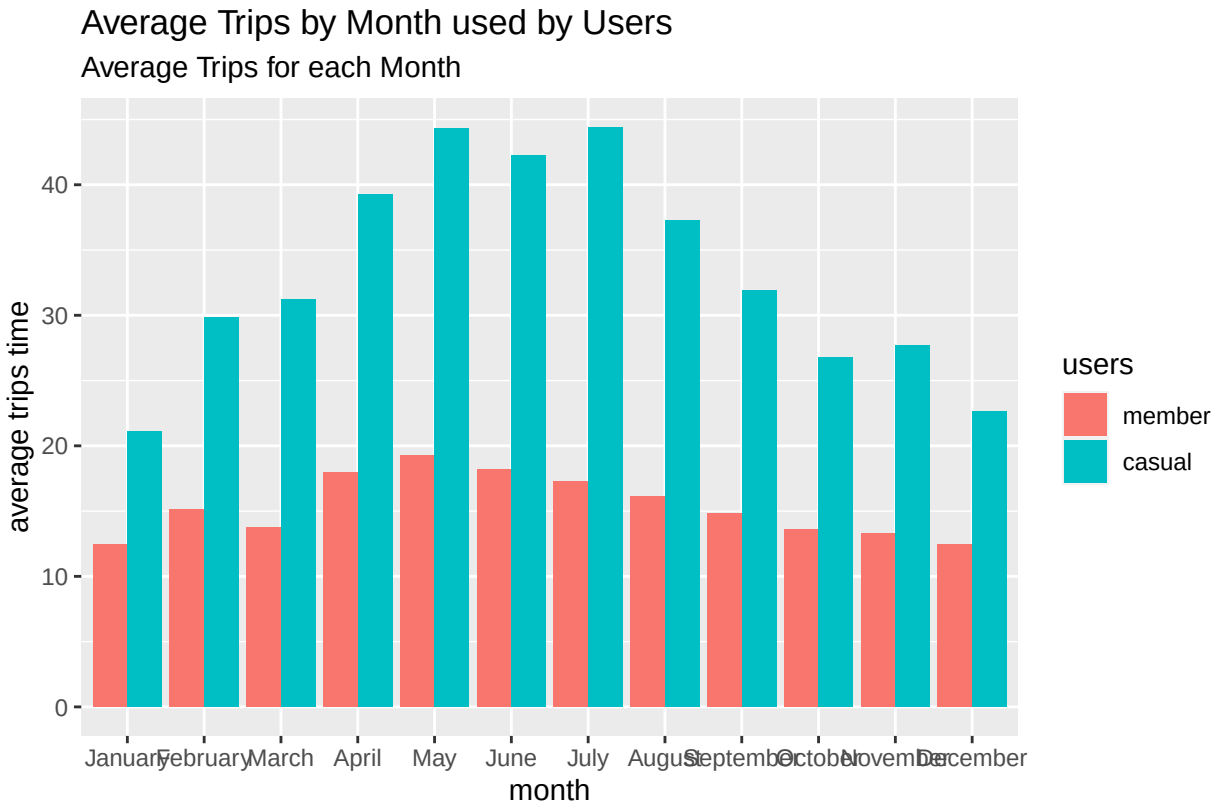


Fig 8

Average duration

The average duration for yearly riders appears to be pretty consistent throughout the year for Casual and also the average time for casual rides is longer than for membership riders having the highest average trips between May and July.

Total trips for each month

```
monthly_ride %>%
 ggplot(aes(month, total_time_month, fill = users))+
 geom_col(position = "dodge")+
 scale_y_continuous()+
```

```
  labs(
    title = "Total Trips for each Month used by Users",
    subtitle = "Total Trips for each of the Month",
    caption = "Fig 8",
    x = "month",
    y = "total trips time"
  )+
  theme()
```

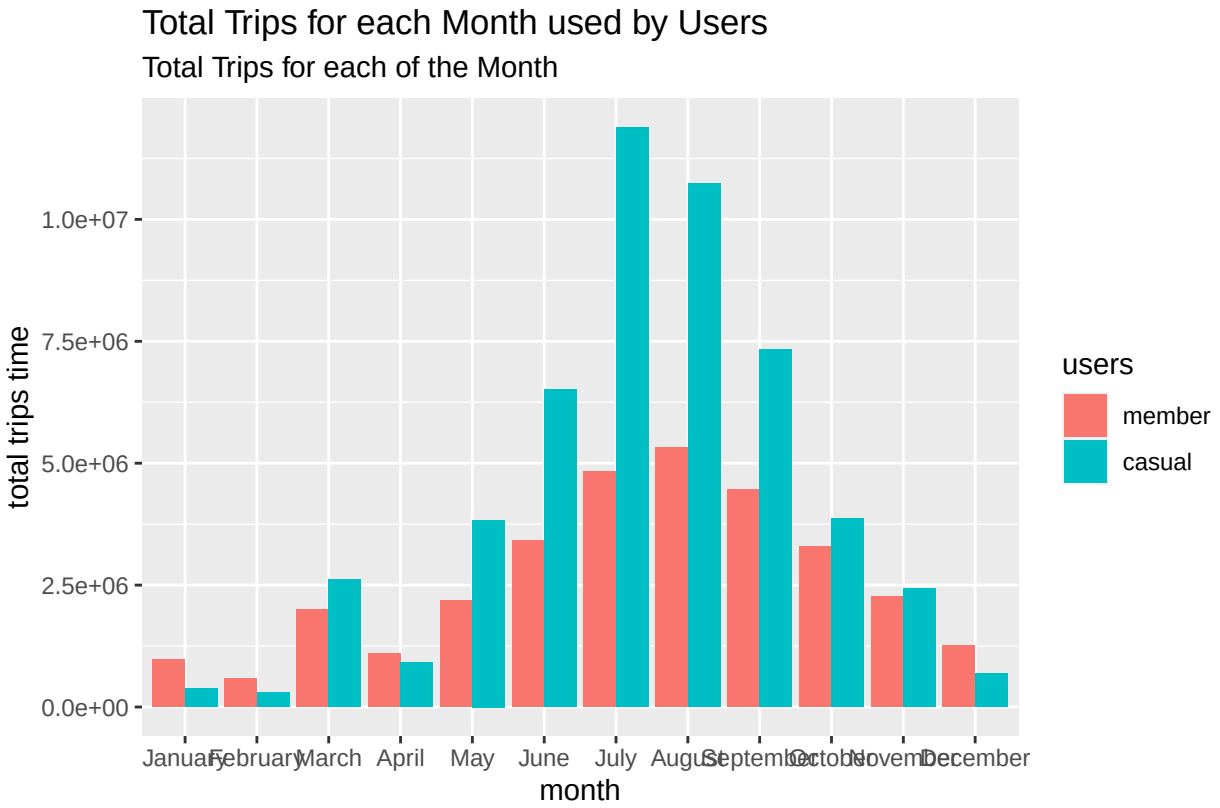


Fig 8

Total trip for each month

From the graph above, all through the month Casual riders are dominant and have the highest total trip time between July and August.

Visualisation and Analysis of trips duration by days of the year

```
daily_ride <- data_time %>%
  group_by(
    users, day
  ) %>%
  summarise(
    num_trips_day = n(),
```

```
    avg_duration_day = mean(trip_time),
    total_duration_day = sum(trip_time)
  )
```

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

Number of trips of length or duration by day and segmented by users

```
daily_ride %>%
  ggplot(aes(day, num_trips_day, fill = users))+
  geom_col(position = "dodge")+
  scale_y_continuous()+
  labs(
    title = "Number of Trips of duration per day used by Users",
    subtitle = "Number of trips of duration per day",
    caption = "Fig 9",
    x = "number of trips",
    y = "day"
  )+
  theme()
```

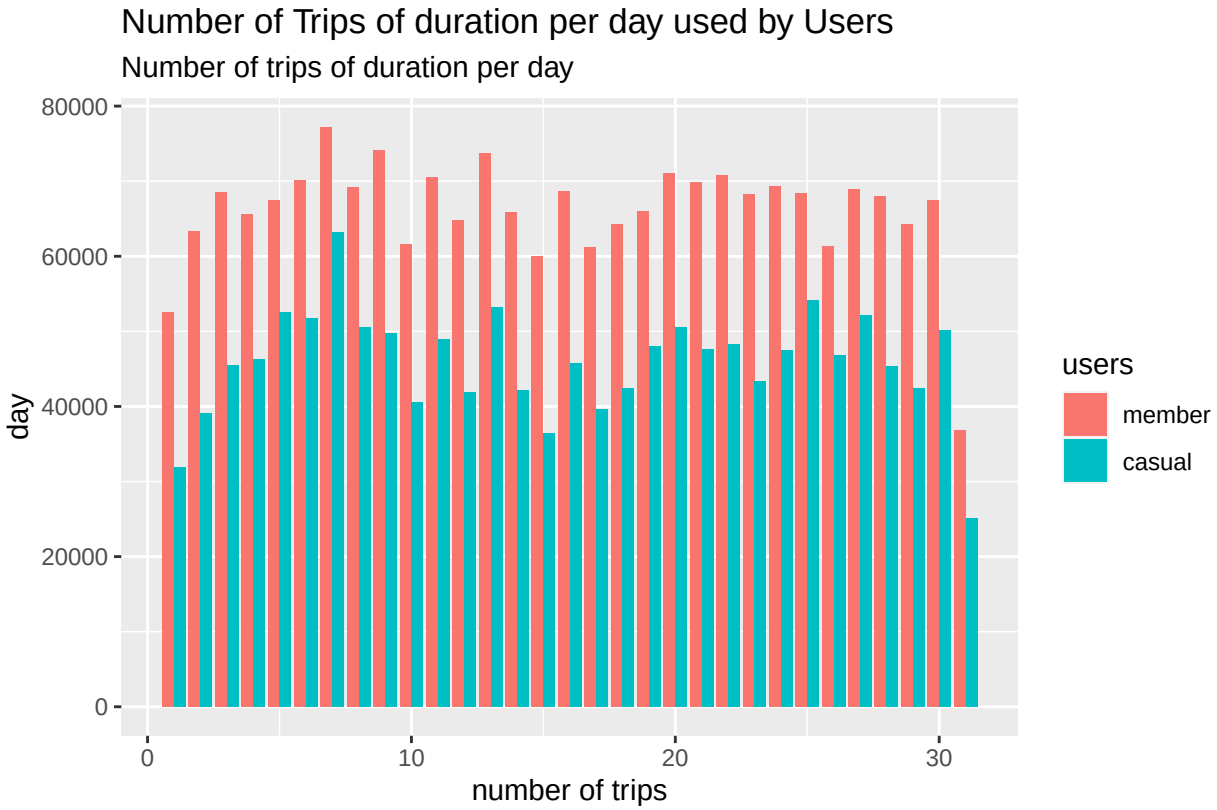


Fig 9

Average trip duration by day and segmented by users

```
daily_ride %>%
  ggplot(aes(day, avg_duration_day, fill = users))+
  geom_col(position = "dodge")+
  scale_y_continuous()+
  labs(
    title = "Average Trip duration per day used by Users",
    subtitle = "Duration of trips per day",
    caption = "Fig 10",
    x = "avg trip duration",
    y = "day"
  )+
  theme()
```

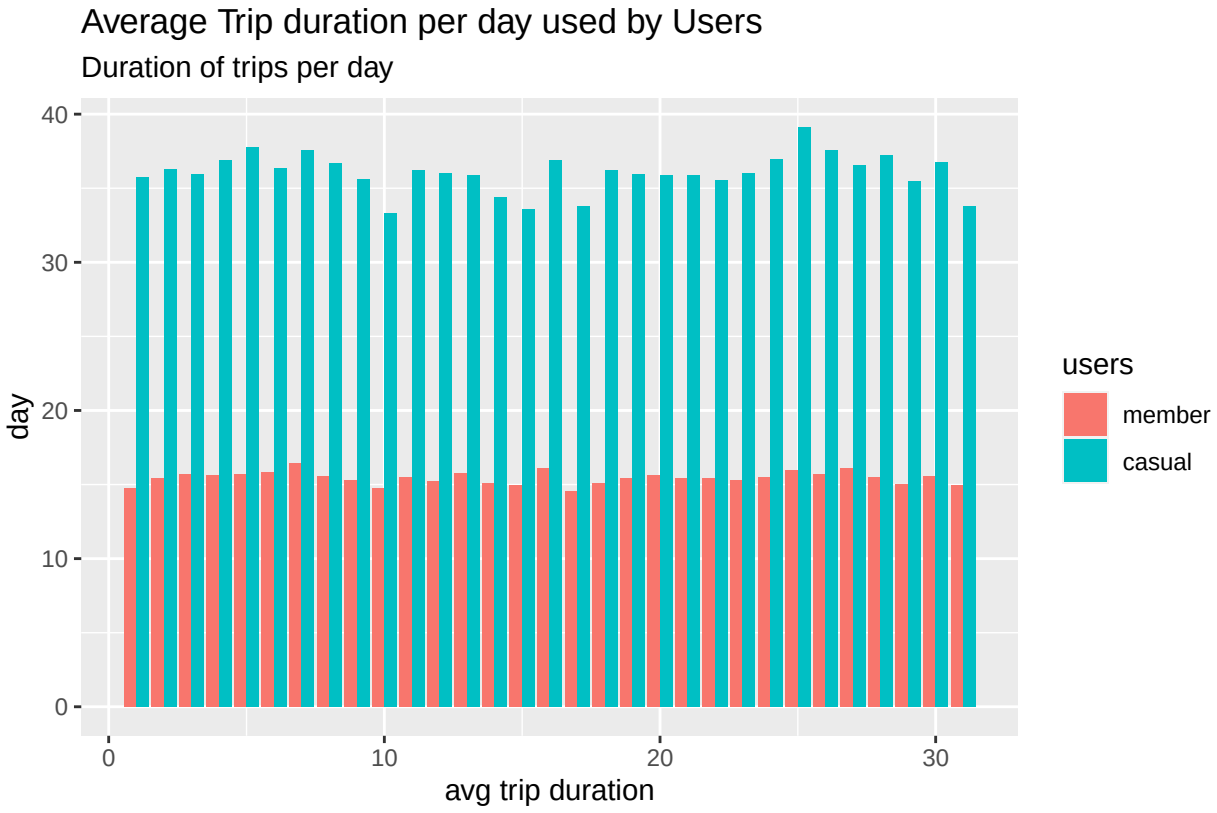


Fig 10

Total trip duration by day and segmented by users

```
daily_ride %>%
  ggplot(aes(day, total_duration_day, fill = users))+
  geom_col(position = "dodge")+
  scale_y_continuous()+
  labs(
```

```
    title = "Total Trip Duration per day used by Users",
    subtitle = "Total trip duration per day",
    caption = "Fig 11",
    x = "total trip duration",
    y = "day"
  )+
  theme()
```

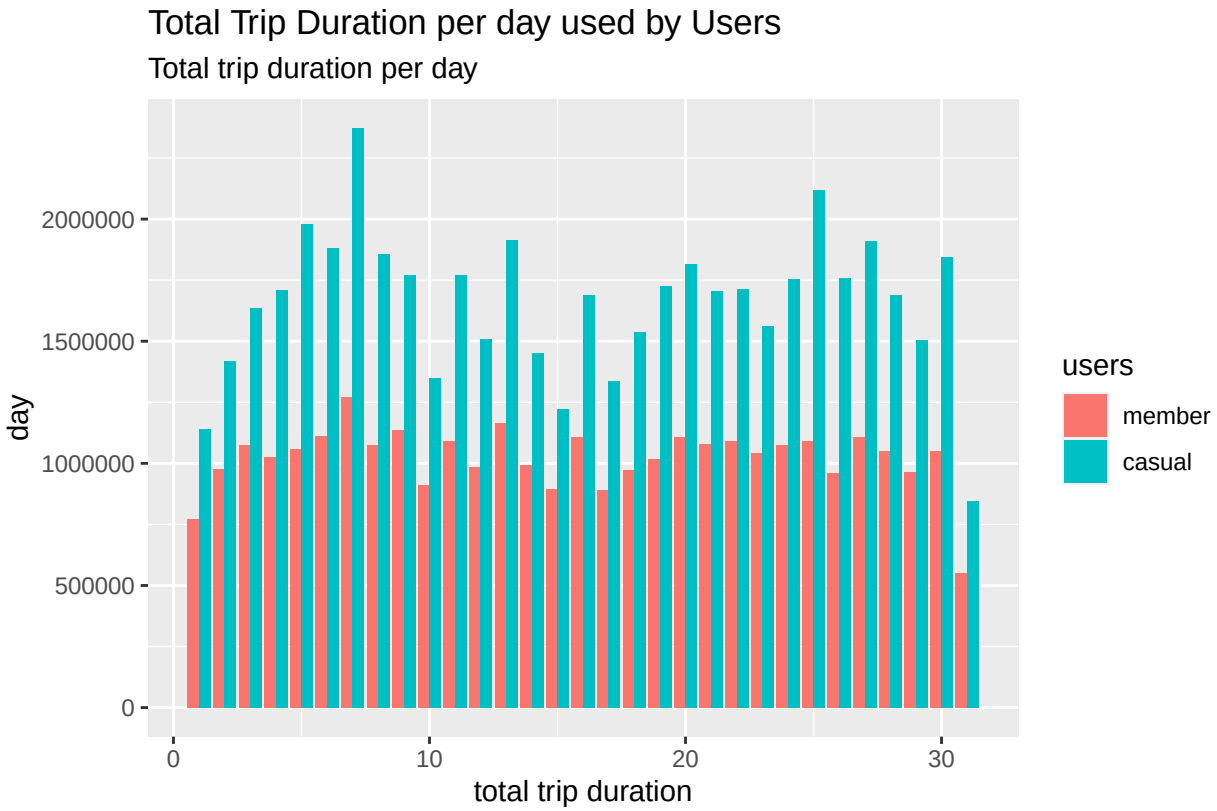


Fig 11

Analysis for trips duration on weeks of the year and segmented by users

```
weekly_ride <- data_time %>%
  group_by(
    users, week
  ) %>%
  summarise(
    num_trips_week = n(),
    avg_tdw = mean(trip_time),
    total_tdw = sum(trip_time)
  )
```

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

Numbers of trips per week of year

```
weekly_ride %>%
  ggplot(aes(week, num_trips_week, fill = users))+
  geom_col(position = "dodge")+
  scale_y_continuous()+
  labs(
    title = "Number of trips per week used by Users",
    subtitle = "Number of Trips per week",
    caption = "Fig 12",
    x = "number of trips",
    y = "week"
  )+
  theme()
```

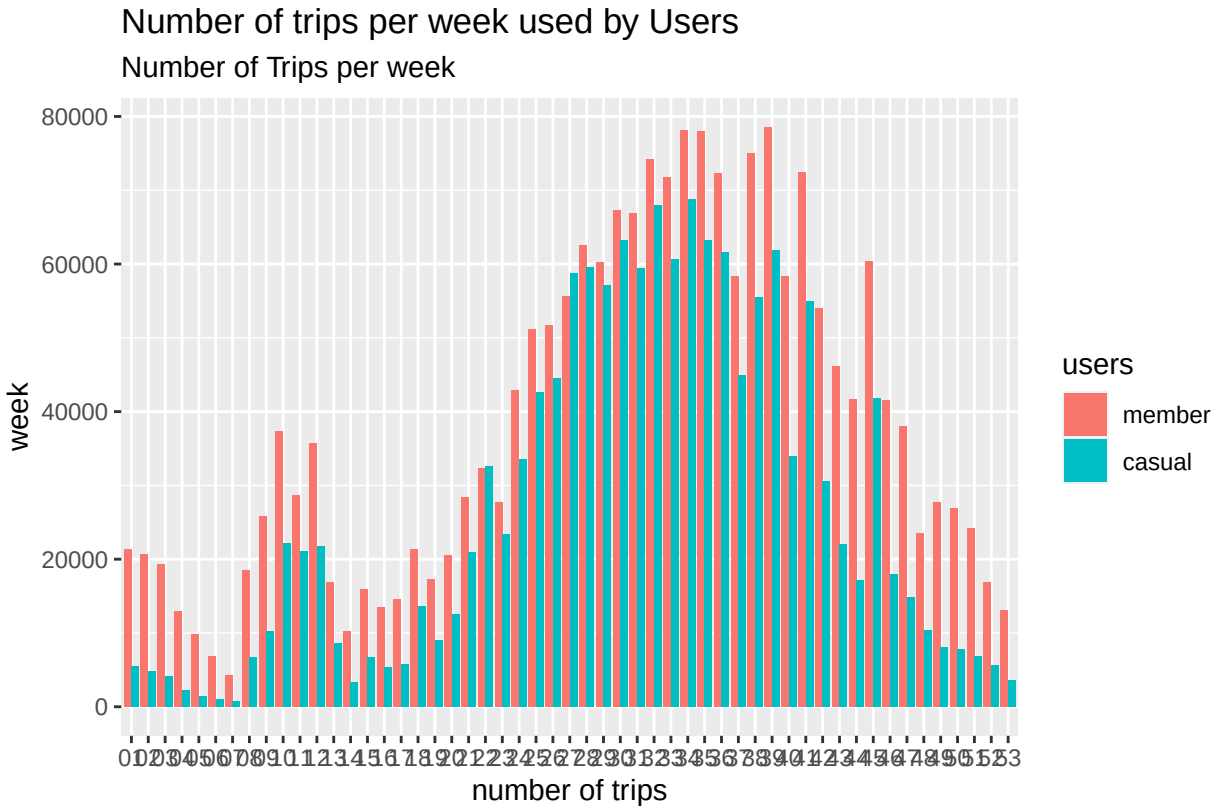


Fig 12

Average trip duration per week of year

```
weekly_ride %>%
  ggplot(aes(week, avg_tdw, fill = users))+
  geom_col(position = "dodge")+
  scale_y_continuous()+
  labs(
```

```
    title = "Average number of trip duration per week used by Users",
    subtitle = "Average number of trips duration per week",
    caption = "Fig 13",
    x = "average number trip duration",
    y = "week"
  )+
  theme()
```

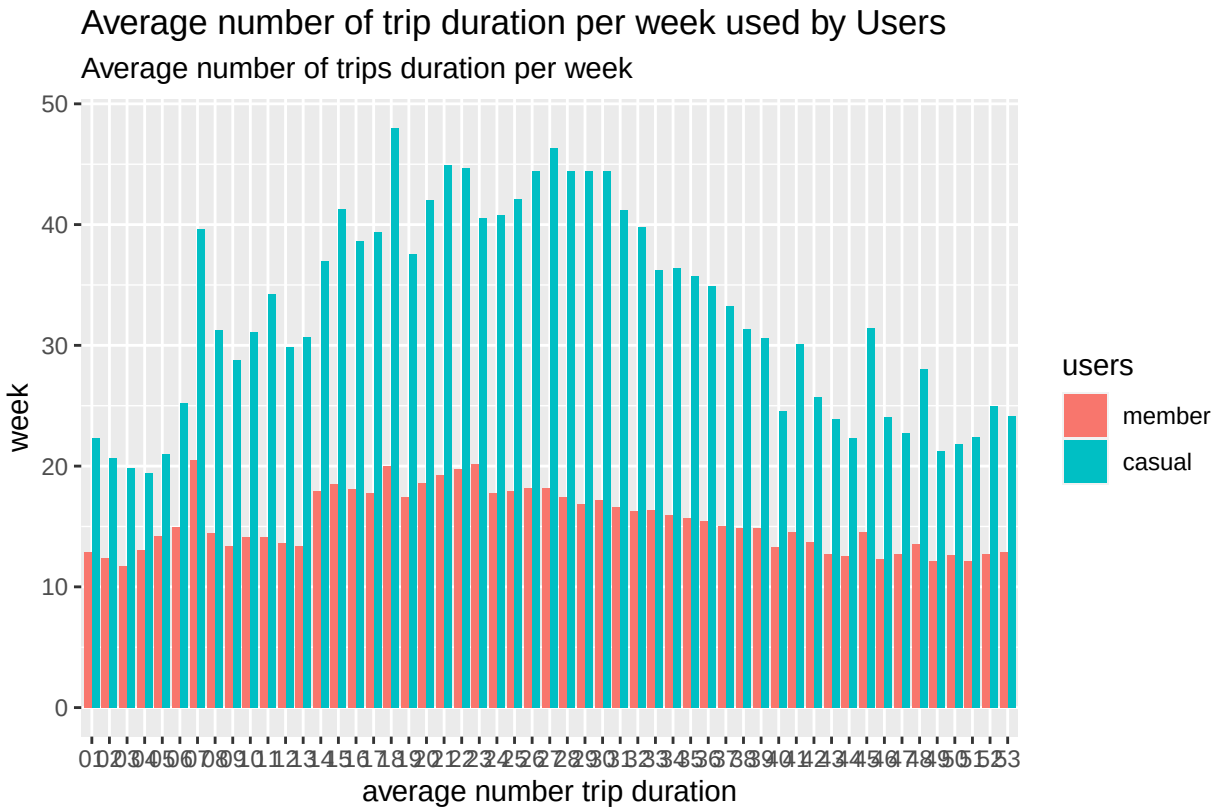


Fig 13

Total trips duration per week of the year

```
weekly_ride %>%
  ggplot(aes(week, total_tdw, fill = users))+
  geom_col(position = "dodge")+
  scale_y_continuous()+
  labs(
    title = "Total trip duration per week used by Users",
    subtitle = "Total trips duration per week",
    caption = "Fig 14",
    x = "total trip duration",
    y = "week"
  )+
  theme()
```

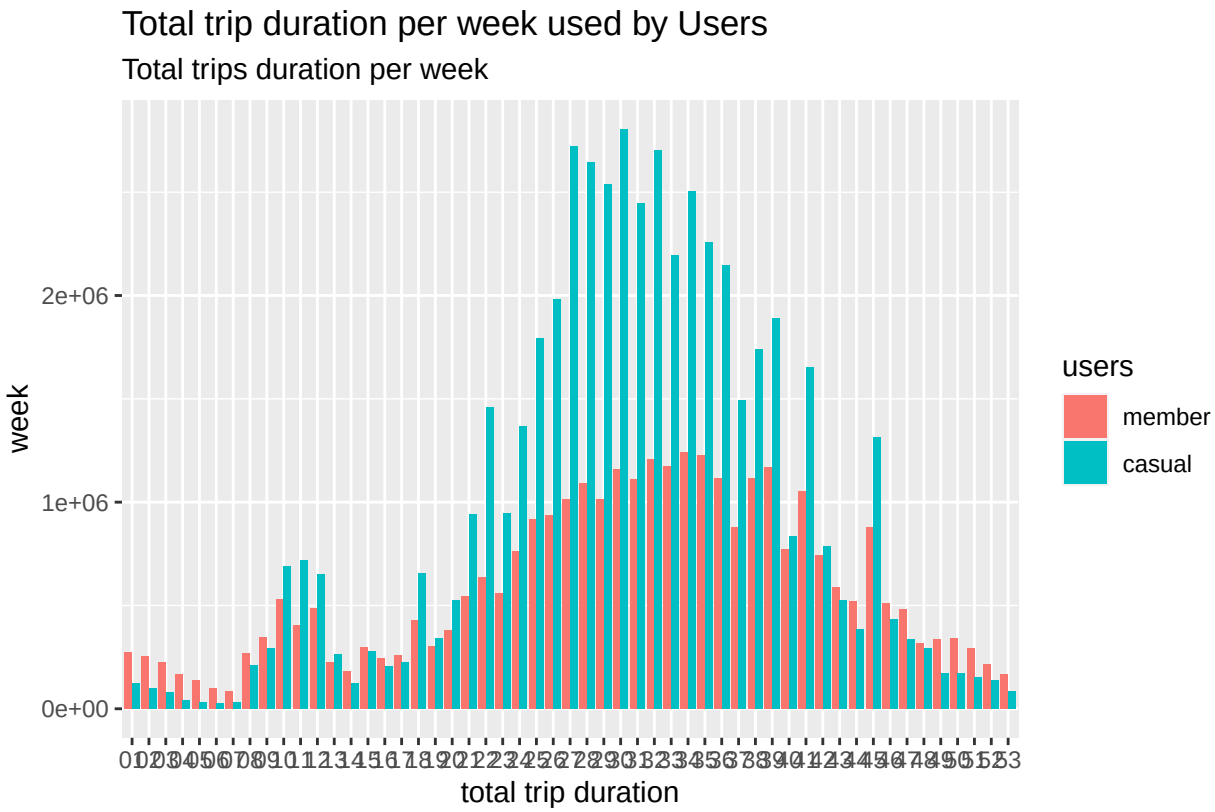


Fig 14

Analysis based on type of bykes

number of rides by bike type

```
types_bikes <- data_time %>%
  group_by(
    users, bikes
  ) %>%
  summarise(
    num_bikes_ride = n(),
    avg_tdw = mean(trip_time),
    total_tdw = sum(trip_time)
  )
```

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

```
types_bikes %>%
  ggplot(aes(bikes,num_bikes_ride, fill = users))+
  geom_col(position = "dodge")+
  scale_y_continuous()+
  labs(
    title = "Number of Trips per Bike Type used by Users",
```

```
    subtitle = "Number of trips per bike type",
    caption = "Fig 15",
    x = "bike type",
    y = "number of trips"
  )+
  theme()
```

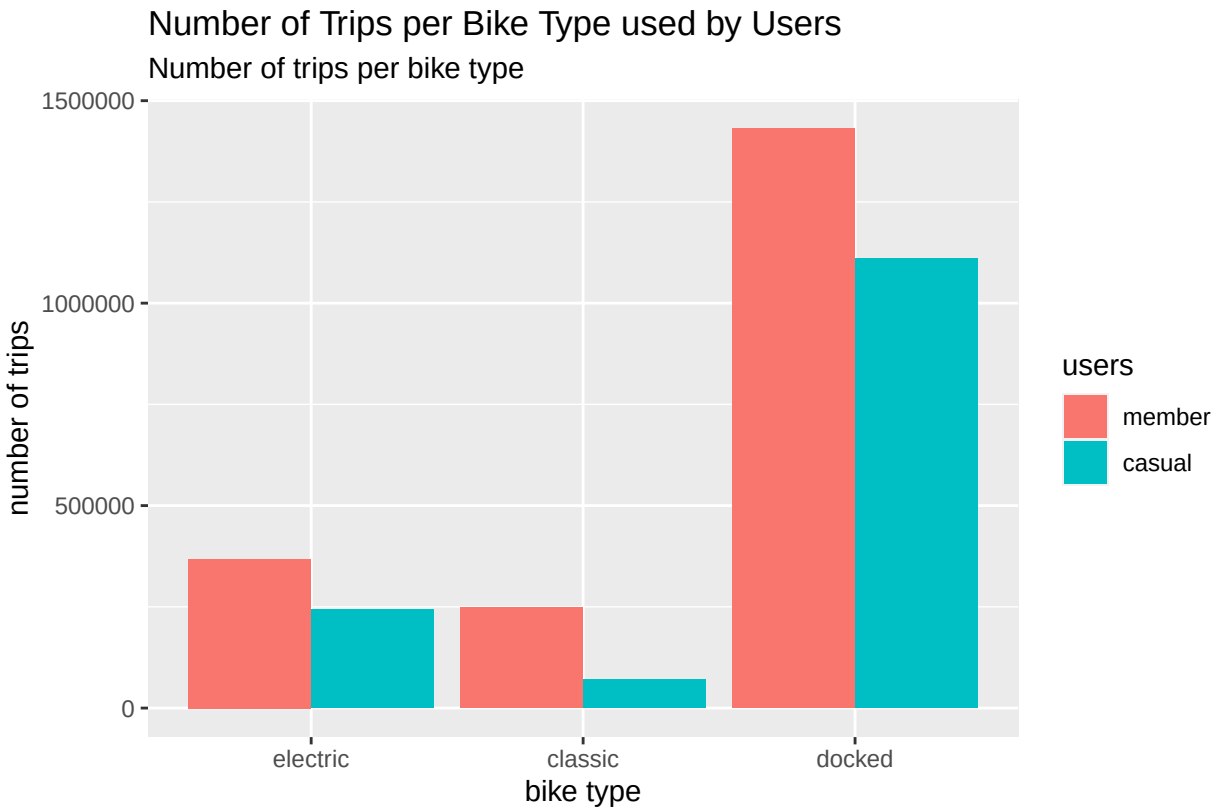


Fig 15

Conclusions:

Docked bikes are the most popular among both riders, followed by electric cycles. The Classic bike appears to be least used bike by casual and member Cyclist rider

Analysis on bike Location

Let preview the data

```
head(data_clean)
```

```
##            ride_id         start_station_name end_station_name start_lat
## 1 E19E6F1B8D4C42ED California Ave & Cortez St                   41.90034
## 2 DC88F20C2C55F27F California Ave & Cortez St                   41.90033
## 3 EC45C94683FE3F27 California Ave & Cortez St                   41.90031
```

```
## 4 4FA453A75AE377DB California Ave & Cortez St                  41.90040
## 5 BE5E8EB4E7263A0B California Ave & Cortez St                  41.90033
## 6 5D8969F88C773979 California Ave & Cortez St                  41.90041
##   start_lng end_lat end_lng  users       trip_time
## 1 -87.69674   41.89  -87.72 member 10.4166667 mins
## 2 -87.69671   41.90  -87.69 member  4.0666667 mins
## 3 -87.69664   41.90  -87.70 member  1.3333333 mins
## 4 -87.69666   41.92  -87.69 member 11.7000000 mins
## 5 -87.69670   41.90  -87.70 casual  0.7166667 mins
## 6 -87.69676   41.94  -87.71 casual 53.7833333 mins
```

Details of the columns of the data frame

```
colnames(data_clean)
```

```
## [1] "ride_id"            "start_station_name" "end_station_name"
## [4] "start_lat"          "start_lng"          "end_lat"
## [7] "end_lng"            "users"              "trip_time"
```

```
bike_station <- data_clean %>%
  group_by(
    users, start_station_name, start_lat, start_lng
  ) %>%
  summarise(
    num_rides_start = n()
  ) %>%
  arrange(-num_rides_start)
```

```
## `summarise()` has grouped output by 'users', 'start_station_name', 'start_lat'.
## You can override using the `.groups` argument.
```

```
head(bike_station)
```

```
## # A tibble: 6 x 5
## # Groups:   users, start_station_name, start_lat [6]
##   users  start_station_name         start_lat start_lng num_rides_start
##   <fct>  <chr>                          <dbl>     <dbl>           <int>
## 1 casual Streeter Dr & Grand Ave         41.9     -87.6           24127
## 2 casual Lake Shore Dr & Monroe St       41.9     -87.6           18804
## 3 member Clark St & Elm St               41.9     -87.6           15966
## 4 casual Theater on the Lake             41.9     -87.6           13599
## 5 member Theater on the Lake             41.9     -87.6           13594
## 6 member Lake Shore Dr & North Blvd      41.9     -87.6           11713
```

check for NA

```
colSums(is.na(bike_station))
```

```
##              users start_station_name          start_lat          start_lng
##                  0                   0                  0                  0
##    num_rides_start
##                  0
```

The most popular 10 start stations

```
bike_station [1:10, ] %>%
  ggplot(aes(start_station_name, num_rides_start, fill = users))+
  geom_col(position = "dodge")+
  coord_flip()+
  labs(
    title = "Most Popular Start Stations",
    subtitle = "Top 10 most popular start stations",
    caption = "Fig 16",
    x = "station name",
    y = "number of trips"
  )+
  theme()
```

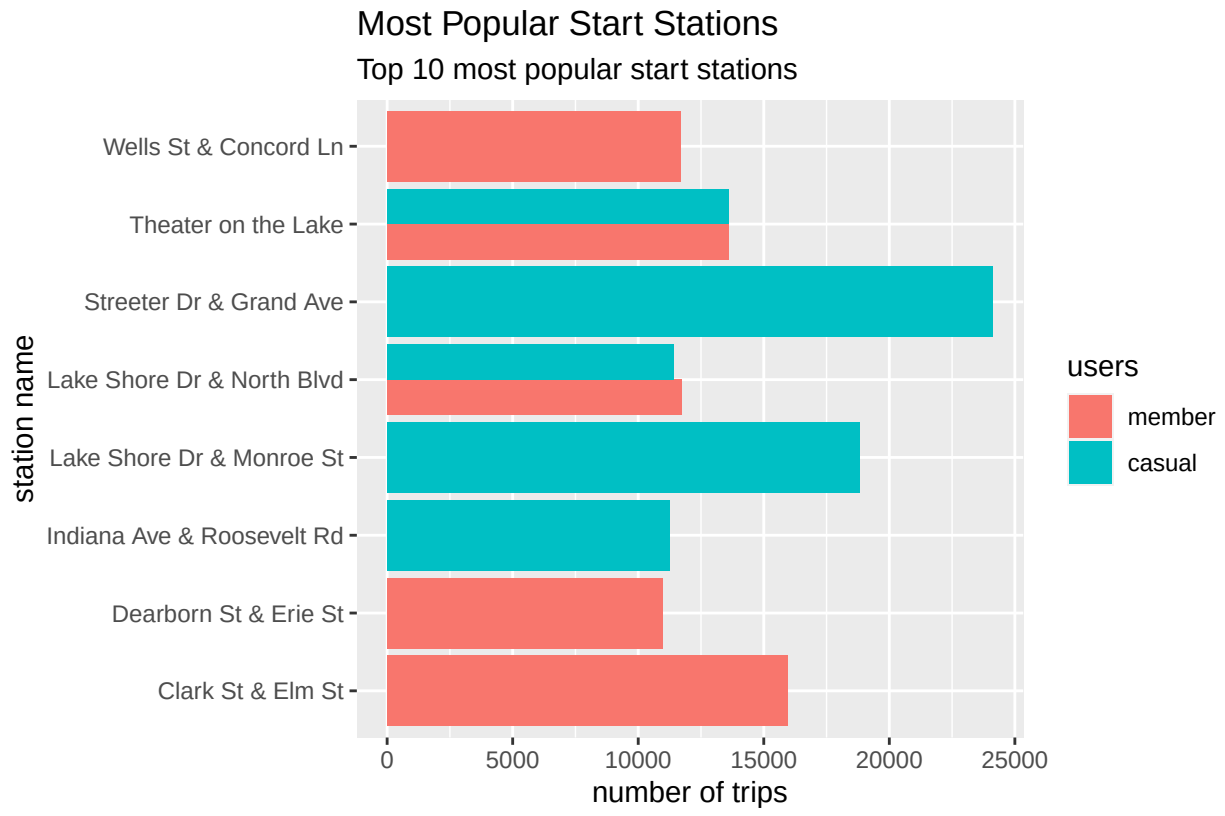


Fig 16

From our chart above it shows that the most popular station is (Streeter Dr & Grand Ave), followed by (Lake shore Dr & Monroe St) and this stations are mostly used by Casual cyclyst rider. The third most

used station is (Clark St & Elm St) and are mostly used by annual member.

The most popular 10 end stations

```
bike_end_station <- data_clean %>%
  group_by(
    users, end_station_name, end_lat, end_lng
  ) %>%
  summarise(
    num_rides_end = n()
  ) %>%
  arrange(-num_rides_end)
```

```
## `summarise()` has grouped output by 'users', 'end_station_name', 'end_lat'. You
## can override using the `.groups` argument.
```

Preview the data frame

```
head(bike_end_station)
```

```
## # A tibble: 6 x 5
## # Groups:   users, end_station_name, end_lat [6]
##   users  end_station_name           end_lat end_lng num_rides_end
##   <fct>  <chr>                        <dbl>   <dbl>         <int>
## 1 casual Streeter Dr & Grand Ave       41.9   -87.6         26465
## 2 casual Lake Shore Dr & Monroe St     41.9   -87.6         18451
## 3 member Clark St & Elm St             41.9   -87.6         16240
## 4 casual Theater on the Lake           41.9   -87.6         15468
## 5 member Theater on the Lake           41.9   -87.6         13425
## 6 casual Lake Shore Dr & North Blvd    41.9   -87.6         12390
```

The most popular 10 end stations

```
bike_end_station[1:10,] %>%
  ggplot(aes(end_station_name, num_rides_end, fill = users))+
  geom_col(position = "dodge")+
  coord_flip()+
  labs(
    title = "Most Popular End Stations used by Users",
    subtitle = "Top 10 most popular end stations",
    caption = "Fig 17",
    x = "station name",
    y = "number of trips"
  )+
  theme()
```

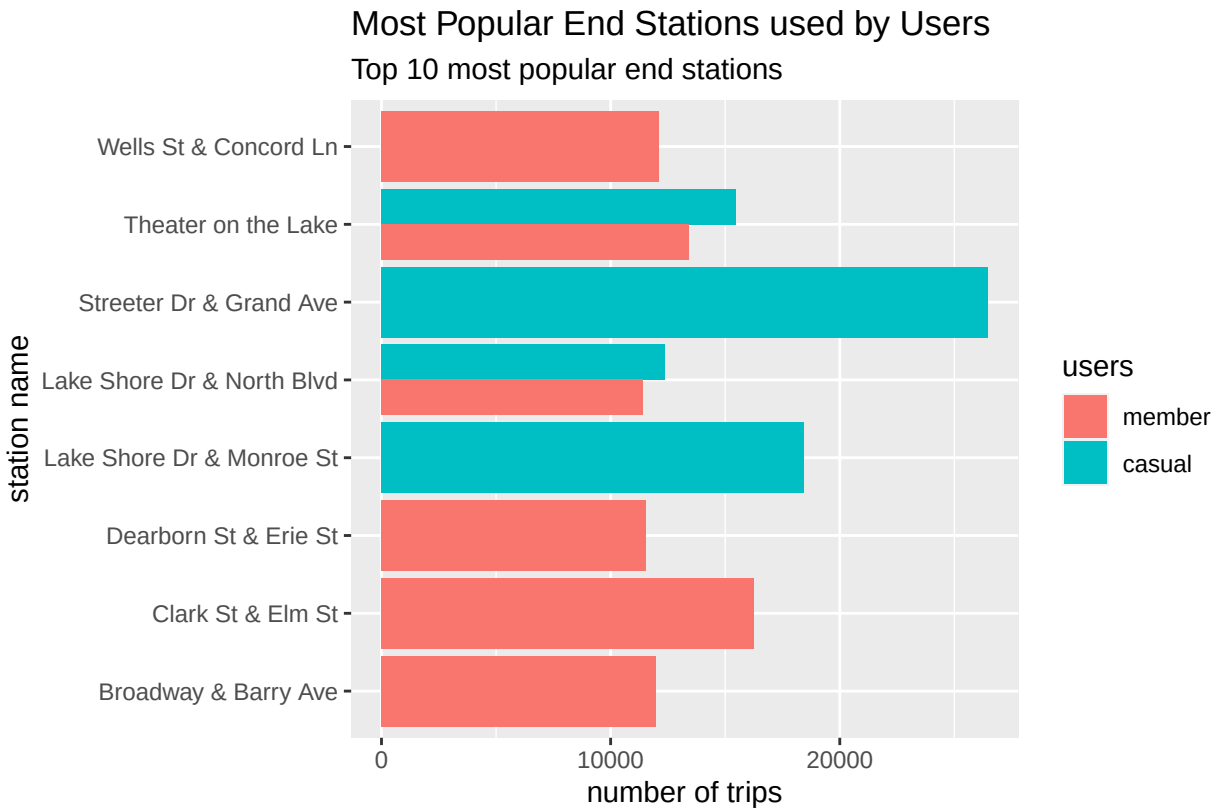


Fig 17

From our chart above it shows that the most popular end station is (Streeter Dr & Grand Ave), followed by (Lake shore Dr & Monroe St) and this stations are mostly used by Casual cyclyst rider.

The third most used station is (Clark St & Elm St) and are mostly used by annual member.

Conclusion

Due to our analysis we found out that the most popular start station and end station are (Streeter Dr & Grand Ave) followed by (Lake shore Dr & Monroe St) and this station are mostly used by casual rider

Most starting and ending stations for Manual cyclists cluster around the other stations.

Recommmendations

- Stations like Streeter Dr & Grand Ave, and Lake shore Dr & Monroe St should be mostly focus on during the marketing Media advert, also targeting dock bike users.
- Most of the rides takes place during the weekend, Marketing campaign should be focused for the busiest casual rider days (Friday, Saturday, and Sunday), busiest hours (afternoon), and most popular months to reach the most riders (June, July and August).
- The preferred bike types by casual riders are dock bike so we need to focus those bike types in the advertisements.