Cyclistic Case Study Oct21

Hezar K

2022-11-29

This is an analysis for Cyclistic Case Study for Google Data Analytics Course. This is an analysis for October 2021.

STEP ONE: INSTALL REQUIRED PACKAGES AND IMPORT DATA

Install the required packages. **Tidyverse** package to import and wrangling the data and **ggplot2** package for visualization of the data. **Lubridate** package for date parsing and **anytime** package for the datetime conversion.

- install.packages("tidyverse")
- install.packages("ggplot2")
- install.packages("lubridate")
- install.packages("anytime")

```
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──
## ✔ ggplot2 3.4.0      ✔ purrr   0.3.5
## ✔ tibble  3.1.8      ✔ dplyr   1.0.10
## ✔ tidyr   1.2.1      ✔ stringr 1.4.1
## ✔ readr   2.1.3      ✔ forcats 0.5.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
## Loading required package: timechange
##
## Attaching package: 'lubridate'
##
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(data.table)
```

```
##
## Attaching package: 'data.table'
##
## The following objects are masked from 'package:lubridate':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
##
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
##
## The following object is masked from 'package:purrr':
##
##     transpose
```

```
library(ggplot2)
library(anytime)
```

Import data from local drive.

```
Oct21 <- read_csv("C:/Users/theby/Documents/202110-divvy-tripdata.csv")
```

```
## Rows: 631226 Columns: 13
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## dttm (2): started_at, ended_at
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

STEP TWO: EXAMINE THE DATA

Examine the dataframe for an overview of the data. Review column names, **colnames()**, dimensions of the dataframe by row and column, **dim()**, the first, **head()**, and the last, **tail()**, six rows in the dataframe, the summary, **summary()**, statistics on the columns of the dataframe, and review the data type structure of columns, **str()**.

View(Oct21)

```
colnames(Oct21)
```

```
##  [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"
```

```
nrow(Oct21)
```

```
## [1] 631226
```

```
dim(Oct21)
```

```
## [1] 631226     13
```

```
head(Oct21)
```

```
## # A tibble: 6 × 13
##   ride_id        ridea…¹ started_at          ended_at            start…² start…³
##   <chr>          <chr>   <dttm>              <dttm>              <chr>   <chr>
## 1 620BC6107255B… electr… 2021-10-22 12:46:42 2021-10-22 12:49:50 Kingsb… KA1503…
## 2 4471C70731AB2… electr… 2021-10-21 09:12:37 2021-10-21 09:14:14 <NA>    <NA>
## 3 26CA69D43D15E… electr… 2021-10-16 16:28:39 2021-10-16 16:36:26 <NA>    <NA>
## 4 362947F0437E1… electr… 2021-10-16 16:17:48 2021-10-16 16:19:03 <NA>    <NA>
## 5 BB731DE2F2EC5… electr… 2021-10-20 23:17:54 2021-10-20 23:26:10 <NA>    <NA>
## 6 7176307BBC097… electr… 2021-10-21 16:57:37 2021-10-21 17:11:58 <NA>    <NA>
## # … with 7 more variables: end_station_name <chr>, end_station_id <chr>,
## #   start_lat <dbl>, start_lng <dbl>, end_lat <dbl>, end_lng <dbl>,
## #   member_casual <chr>, and abbreviated variable names ¹rideable_type,
## #   ²start_station_name, ³start_station_id
```

```
tail(Oct21)
```

```
## # A tibble: 6 × 13
##   ride_id        ridea…¹ started_at          ended_at            start…² start…³
##   <chr>          <chr>   <dttm>              <dttm>              <chr>   <chr>
## 1 817A854B4429A… classi… 2021-10-15 18:01:23 2021-10-15 18:09:41 Frankl… TA1305…
## 2 BA077FDD42DAB… classi… 2021-10-14 21:45:05 2021-10-14 22:07:25 Frankl… 13017
## 3 B7D99254E798A… classi… 2021-10-02 15:28:28 2021-10-02 15:51:02 Street… 13022
## 4 BCCFD66DA4664… electr… 2021-10-08 16:47:10 2021-10-08 16:52:43 Calume… 15546
## 5 623E0F6F50CDD… classi… 2021-10-08 07:49:47 2021-10-08 07:55:15 Calume… 15546
## 6 83FA6AC52B7B7… classi… 2021-10-02 12:55:45 2021-10-02 13:21:10 Winthr… TA1308…
## # … with 7 more variables: end_station_name <chr>, end_station_id <chr>,
## #   start_lat <dbl>, start_lng <dbl>, end_lat <dbl>, end_lng <dbl>,
## #   member_casual <chr>, and abbreviated variable names ¹rideable_type,
## #   ²start_station_name, ³start_station_id
```

```
summary(Oct21)
```

```
##    ride_id          rideable_type        started_at                    
##  Length:631226      Length:631226      Min.   :2021-10-01 00:00:09.00  
##  Class :character   Class :character   1st Qu.:2021-10-08 12:25:58.25  
##  Mode  :character   Mode  :character   Median :2021-10-15 05:31:57.00  
##                                        Mean   :2021-10-15 08:38:27.35  
##                                        3rd Qu.:2021-10-21 19:25:00.75  
##                                        Max.   :2021-10-31 23:59:49.00  
##                                                                        
##     ended_at                      start_station_name start_station_id  
##  Min.   :2021-10-01 00:03:11.00   Length:631226      Length:631226     
##  1st Qu.:2021-10-08 12:46:34.00   Class :character   Class :character  
##  Median :2021-10-15 05:56:26.50   Mode  :character   Mode  :character  
##  Mean   :2021-10-15 08:57:32.92                                        
##  3rd Qu.:2021-10-21 19:37:25.00                                        
##  Max.   :2021-11-03 21:45:48.00                                        
##                                                                        
##  end_station_name   end_station_id       start_lat       start_lng     
##  Length:631226      Length:631226      Min.   :41.65   Min.   :-87.83  
##  Class :character   Class :character   1st Qu.:41.88   1st Qu.:-87.66  
##  Mode  :character   Mode  :character   Median :41.90   Median :-87.64  
##                                        Mean   :41.90   Mean   :-87.65  
##                                        3rd Qu.:41.93   3rd Qu.:-87.63  
##                                        Max.   :42.07   Max.   :-87.52  
##                                                                        
##     end_lat         end_lng       member_casual     
##  Min.   :41.60   Min.   :-87.96   Length:631226     
##  1st Qu.:41.88   1st Qu.:-87.66   Class :character  
##  Median :41.90   Median :-87.64   Mode  :character  
##  Mean   :41.90   Mean   :-87.65                     
##  3rd Qu.:41.93   3rd Qu.:-87.63                     
##  Max.   :42.13   Max.   :-87.52                     
##  NA's   :484     NA's   :484
```

```
str(Oct21)
```

```
## spc_tbl_ [631,226 × 13] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:631226] "620BC6107255BF4C" "4471C70731AB2E45" "26CA69D43D15EE14" "362947F0437E15
14" ...
##  $ rideable_type     : chr [1:631226] "electric_bike" "electric_bike" "electric_bike" "electric_bike" ...
##  $ started_at        : POSIXct[1:631226], format: "2021-10-22 12:46:42" "2021-10-21 09:12:37" ...
##  $ ended_at          : POSIXct[1:631226], format: "2021-10-22 12:49:50" "2021-10-21 09:14:14" ...
##  $ start_station_name: chr [1:631226] "Kingsbury St & Kinzie St" NA NA NA ...
##  $ start_station_id  : chr [1:631226] "KA1503000043" NA NA NA ...
##  $ end_station_name  : chr [1:631226] NA NA NA NA ...
##  $ end_station_id    : chr [1:631226] NA NA NA NA ...
##  $ start_lat         : num [1:631226] 41.9 41.9 41.9 41.9 41.9 ...
##  $ start_lng         : num [1:631226] -87.6 -87.7 -87.7 -87.7 -87.7 ...
##  $ end_lat           : num [1:631226] 41.9 41.9 41.9 41.9 41.9 ...
##  $ end_lng           : num [1:631226] -87.6 -87.7 -87.7 -87.7 -87.7 ...
##  $ member_casual     : chr [1:631226] "member" "member" "member" "member" ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   ride_id = col_character(),
##   ..   rideable_type = col_character(),
##   ..   started_at = col_datetime(format = ""),
##   ..   ended_at = col_datetime(format = ""),
##   ..   start_station_name = col_character(),
##   ..   start_station_id = col_character(),
##   ..   end_station_name = col_character(),
##   ..   end_station_id = col_character(),
##   ..   start_lat = col_double(),
##   ..   start_lng = col_double(),
##   ..   end_lat = col_double(),
##   ..   end_lng = col_double(),
##   ..   member_casual = col_character()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

Create new columns as for *date*, *month*, *day*, *year*, *day_of_week*, and *ride_length* in seconds.

```
Oct21$date <- as.Date(Oct21$started_at)
Oct21$month <- format(as.Date(Oct21$date), "%m")
Oct21$day <- format(as.Date(Oct21$date), "%d")
Oct21$year <- format(as.Date(Oct21$date), "%Y")
Oct21$day_of_week <- format(as.Date(Oct21$date), "%A")
Oct21$ride_length <- difftime(Oct21$ended_at,Oct21$started_at)
```

Convert *ride_length* column to numeric in order to run calculations on the data. First, check to see if the data type is numeric, and then convert if needed.

```
is.numeric(Oct21$ride_length)
```

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

Recheck *ride_length* data type.

```
Oct21$ride_length <- as.numeric(as.character(Oct21$ride_length))
is.numeric(Oct21$ride_length)
```

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

STEP THREE: CLEAN DATA

na.omit() will remove all NA from the dataframe.

```
Oct21 <- na.omit(Oct21)
```

Remove rows with the *ride_id* column character length is not 16. This will remove all the scientific ride ids that we noticed while examining the data.

```
Oct21 <- subset(Oct21, nchar(as.character(ride_id)) == 16)
```

Remove rows with the *ride_length* less than 1 minute.

```
Oct21 <- subset (Oct21, ride_length > "1")
```

STEP FOUR: ANALYZE DATA

Analyze the dataframe by find the **mean**, **median**, **max** (maximum), and **min** (minimum) of *ride_length*.

```
mean(Oct21$ride_length)
```

```
## [1] 1060.467
```

```
median(Oct21$ride_length)
```

```
## [1] 637
```

```
max(Oct21$ride_length)
```

```
## [1] 2442301
```

```
min(Oct21$ride_length)
```

```
## [1] 2
```

Run a statistical summary of the *ride_length*.

```
summary(Oct21$ride_length)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##       2     371     637    1060    1131 2442301
```

Compare the members and casual users

```
aggregate(Oct21$ride_length ~ Oct21$member_casual, FUN = mean)
```

```
##   Oct21$member_casual Oct21$ride_length
## 1              casual         1578.9908
## 2              member          720.9674
```

```
aggregate(Oct21$ride_length ~ Oct21$member_casual, FUN = median)
```

```
##   Oct21$member_casual Oct21$ride_length
## 1              casual               862
## 2              member               530
```

```
aggregate(Oct21$ride_length ~ Oct21$member_casual, FUN = max)
```

```
##   Oct21$member_casual Oct21$ride_length
## 1              casual           2442301
## 2              member             84908
```

```
aggregate(Oct21$ride_length ~ Oct21$member_casual, FUN = min)
```

```
##   Oct21$member_casual Oct21$ride_length
## 1              casual                 2
## 2              member                 2
```

Aggregate the average ride length by each day of the week for members and users.

```
aggregate(Oct21$ride_length ~ Oct21$member_casual + Oct21$day_of_week, FUN = mean)
```

```
##    Oct21$member_casual Oct21$day_of_week Oct21$ride_length
## 1               casual            Friday         1468.0023
## 2               member            Friday          700.5979
## 3               casual            Monday         1502.0862
## 4               member            Monday          672.3251
## 5               casual          Saturday         1762.5377
## 6               member          Saturday          816.7086
## 7               casual            Sunday         1895.5110
## 8               member            Sunday          821.7609
## 9               casual          Thursday         1109.0232
## 10              member          Thursday          647.8775
## 11              casual           Tuesday         1341.7343
## 12              member           Tuesday          682.2092
## 13              casual         Wednesday         1239.7723
## 14              member         Wednesday          688.9376
```

Sort the days of the week in order.

```
Oct21$day_of_week <- ordered(Oct21$day_of_week, levels=c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday",
"Friday", "Saturday"))
```

Assign the aggregate the average ride length by each day of the week for members and users to x.

```
x <- aggregate(Oct21$ride_length ~ Oct21$member_casual + Oct21$day_of_week, FUN = mean)

head(x)
```

```
##   Oct21$member_casual Oct21$day_of_week Oct21$ride_length
## 1              casual            Sunday         1895.5110
## 2              member            Sunday          821.7609
## 3              casual            Monday         1502.0862
## 4              member            Monday          672.3251
## 5              casual           Tuesday         1341.7343
## 6              member           Tuesday          682.2092
```

Find the average ride length of member riders and casual riders per day and assign it to y.

```
y <- Oct21 %>%
  mutate(weekday = wday(started_at)) %>%
  group_by(member_casual, weekday) %>%
  summarise(number_of_rides = n(),
            average_duration = mean(ride_length), .groups = 'drop') %>%
  arrange(member_casual, weekday)

head(y)
```

```
## # A tibble: 6 × 4
##   member_casual weekday number_of_rides average_duration
##   <chr>           <int>           <int>            <dbl>
## 1 casual              1           40263            1896.
## 2 casual              2           15573            1502.
## 3 casual              3           19043            1342.
## 4 casual              4           18615            1240.
## 5 casual              5           15087            1109.
## 6 casual              6           28527            1468.
```

Analyze the dataframe to find the frequency of member riders, casual riders, classic bikes, docked bikes, and electric bikes.

```
table(Oct21$member_casual)
```

```
## 
## casual member 
## 189099 288815
```

```
table(Oct21$rideable_type)
```

```
## 
##  classic_bike   docked_bike electric_bike 
##        315144         22689        140081
```

```
table(Oct21$day_of_week)
```

```
## 
##    Sunday    Monday   Tuesday Wednesday  Thursday    Friday  Saturday 
##     77097     47919     64333     63554     50769     74257     99985
```

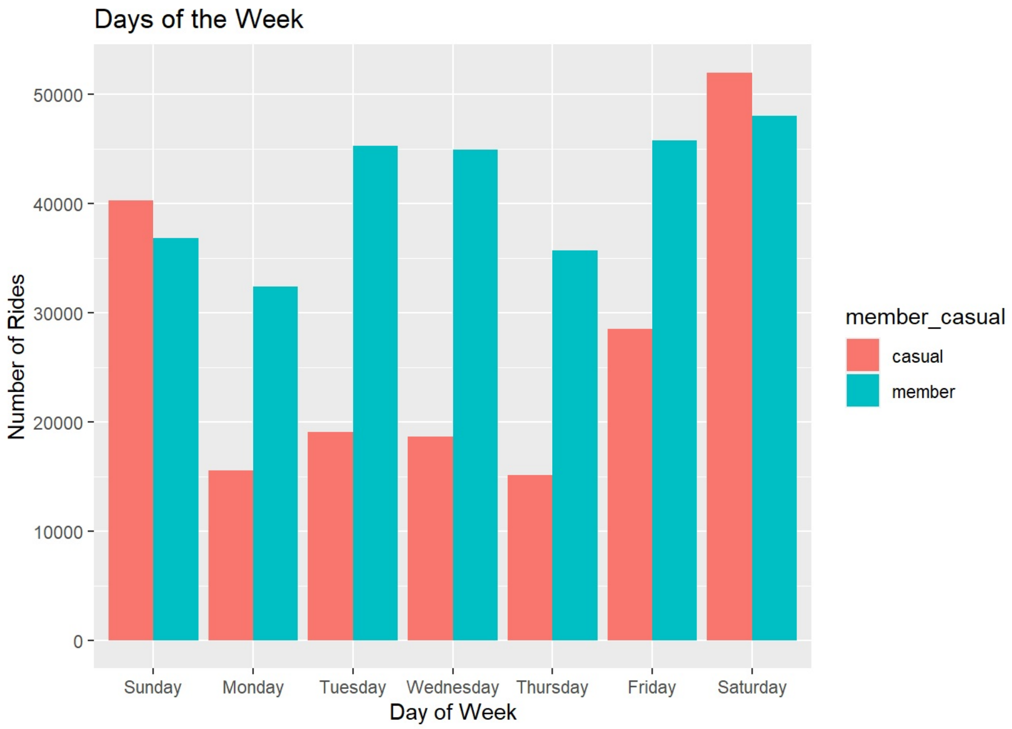
STEP FIVE: VISUALIZATION

Display full digits instead of scientific number.

```
options(scipen=999)
```

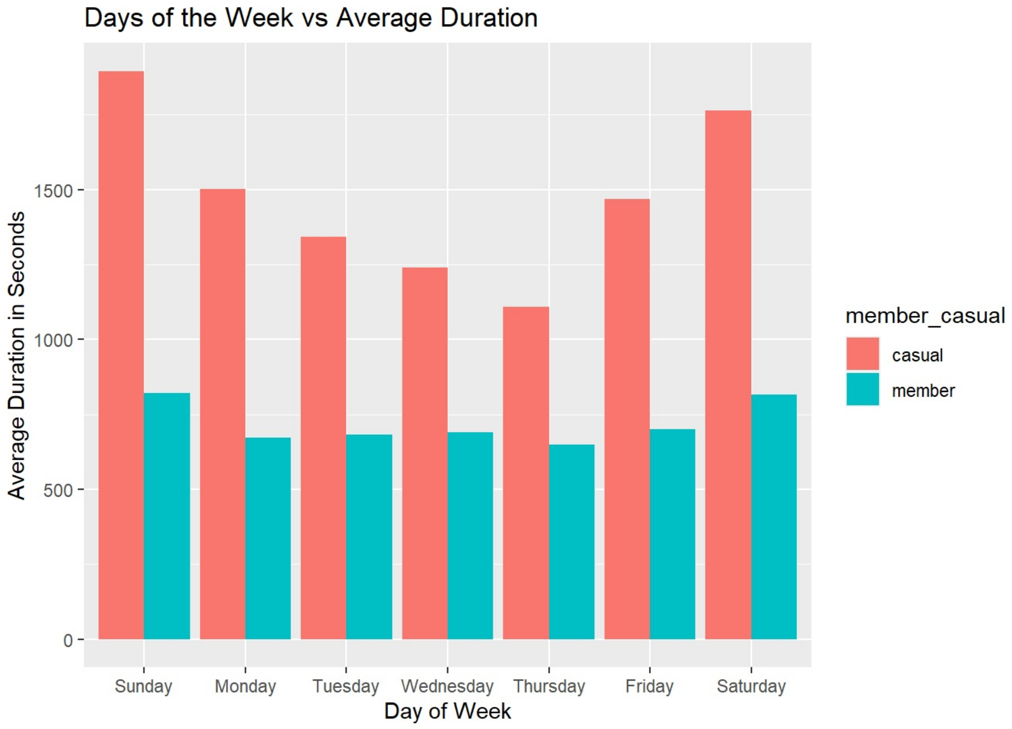
Plot the number of rides by user type during the week.

```
Oct21 %>%
  mutate(day_of_week) %>%
  group_by(member_casual,day_of_week) %>%
  summarise(number_of_rides = n(), average_duration = mean(ride_length), .groups = 'drop') %>%
  arrange(member_casual, day_of_week)  %>%
  ggplot(aes(x = day_of_week, y = number_of_rides, fill = member_casual)) +
  geom_col(position = "dodge")+
labs(x = "Day of Week",
     y= "Number of Rides",
     title= "Days of the Week")
```



Plot the duration of the ride by user type during the week.

```
Oct21 %>%
  mutate(day_of_week) %>%
  group_by(member_casual, day_of_week) %>%
  summarise(number_of_rides = n(), average_duration = mean(ride_length), .groups = 'drop') %>%
  arrange(member_casual, day_of_week)  %>%
  ggplot(aes(x = day_of_week, y = average_duration, fill = member_casual)) +
  geom_col(position = "dodge") +
  labs(x = "Day of Week",
       y= "Average Duration in Seconds",
       title= "Days of the Week vs Average Duration")
```



Create new dataframe for plots for weekday trends vs weekend trends.

```
mc<- as.data.frame(table(Oct21$day_of_week,Oct21$member_casual))
```

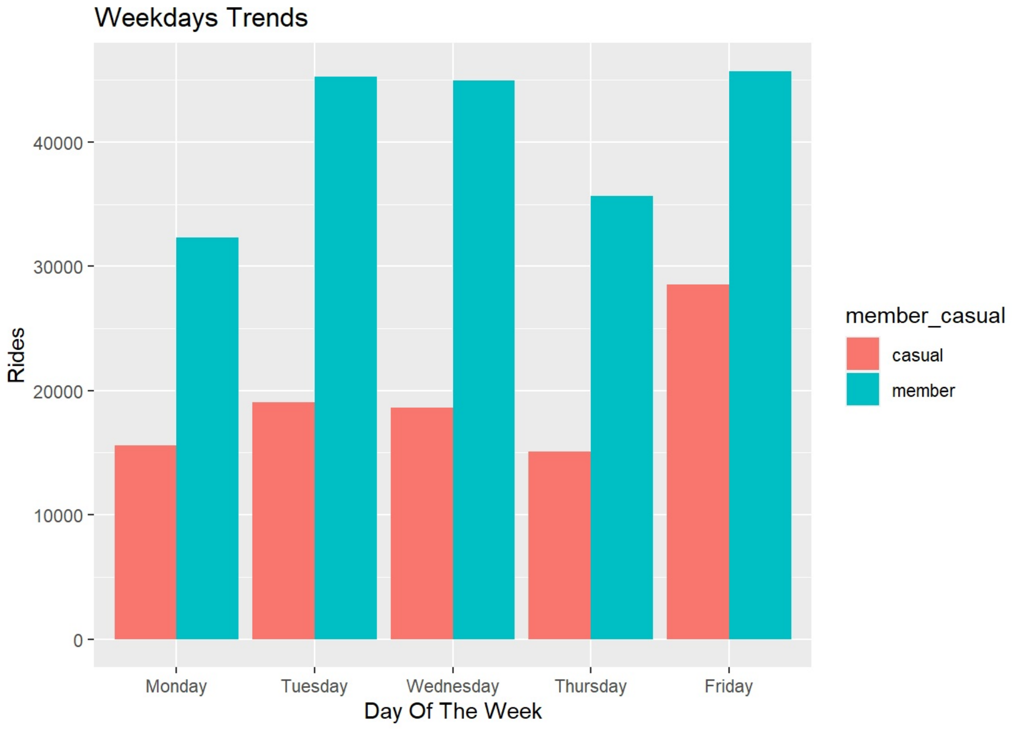
Rename columns

```
mc<-rename(mc, day_of_week = Var1, member_casual = Var2)
head(mc)
```

```
##   day_of_week member_casual  Freq
## 1      Sunday        casual 40263
## 2      Monday        casual 15573
## 3     Tuesday        casual 19043
## 4   Wednesday        casual 18615
## 5    Thursday        casual 15087
## 6      Friday        casual 28527
```

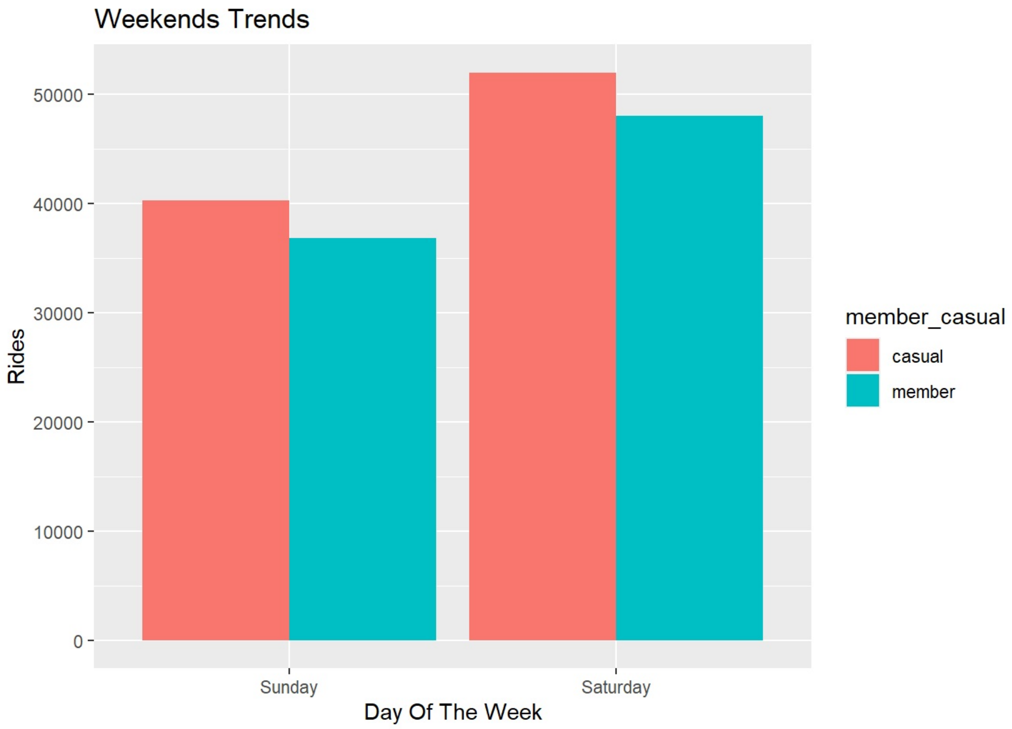
Weekday trends (Monday through Friday).

```
mc %>%
  filter(day_of_week == "Monday" |
           day_of_week == "Tuesday" |
           day_of_week == "Wednesday" |
           day_of_week == "Thursday" |
           day_of_week == "Friday") %>%
  ggplot(aes(x = day_of_week, y = Freq, fill = member_casual))+
  geom_bar(stat = "identity" , position = "dodge") +
  labs(title = "Weekdays Trends",
       x= "Day Of The Week",
       y = "Rides")
```



Weekend trends (Sunday and Saturday).

```
mc %>%
  filter(day_of_week == "Sunday" |
           day_of_week == "Saturday") %>%
  ggplot(aes(x = day_of_week, y = Freq, fill = member_casual))+
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Weekends Trends",
       x= "Day Of The Week",
       y = "Rides")
```



Create dataframe for member and casual riders vs ride type

```
rt<- as.data.frame(table(Oct21$rideable_type,Oct21$member_casual))
```

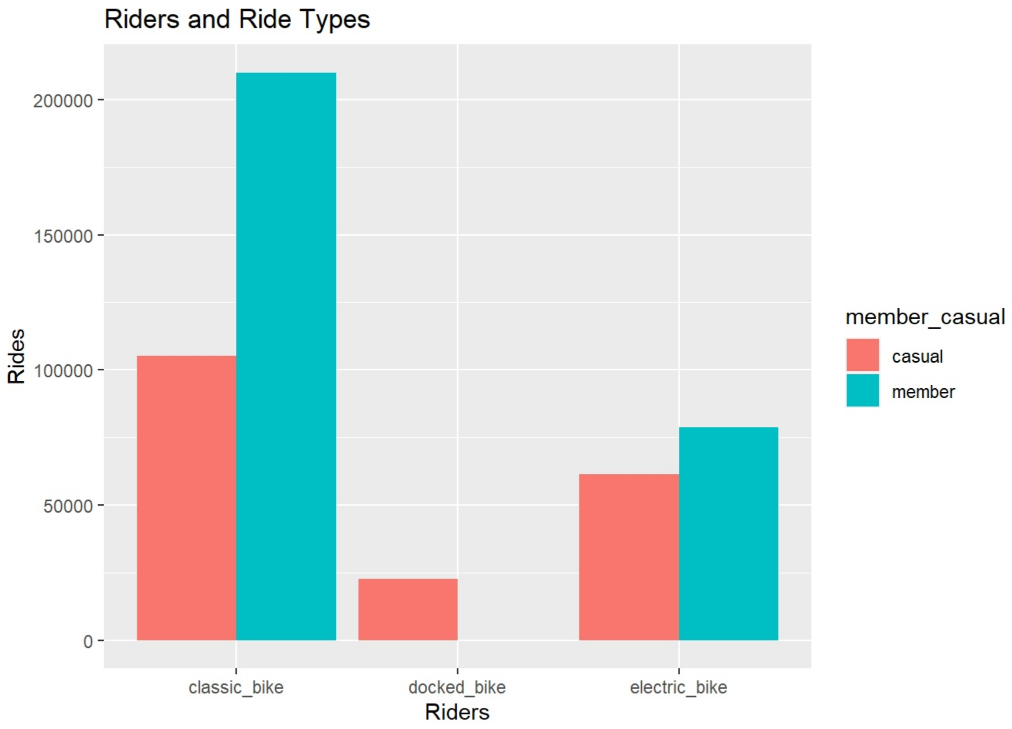
Rename columns.

```
rt<-rename(rt, rideable_type = Var1, member_casual = Var2)
head(rt)
```

```
##   rideable_type member_casual   Freq
## 1  classic_bike        casual 105121
## 2   docked_bike        casual  22689
## 3 electric_bike        casual  61289
## 4  classic_bike        member 210023
## 5   docked_bike        member      0
## 6 electric_bike        member  78792
```

Plot for bike user vs bike type.

```
rt %>%
  filter(member_casual == "member" |
           member_casual == "casual") %>%
  ggplot(aes(x = rideable_type, y = Freq, fill = member_casual))+
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Riders and Ride Types",
       x= "Riders",
       y = "Rides")
```



STEP SIX: EXPORT ANALYZED DATA

Save the analyzed data as a new file. fwrite(Oct21, “Oct21.csv”)